VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Přístupový systém založený na ESP32

Access System Based on ESP32

2019 Lukáš Klein

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Lukáš Klein**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Přístupový systém založený na ESP32
Access System Based on ESP32

Jazyk vypracování: čeština

Zásady pro vypracování:

Pro vývoj software bude k dispozici hardware sestávající z modulu základny a modulu čteček. Oba moduly budou osazeny procesorem založeným na ESP32. Modul základny bude obsahovat komunikační rozhraní Ethernet a RS422. Modul čteček bude obsahovat rozhraní RS422, čtečku karet RFID (125kHz) a NFC (13.56MHz). Hlavní cíl práce bude vytvořit software pro oba moduly a dále software pro server pracující na operačním systému Linux, který bude obsluhovat čtečku NFC. Komunikace se čtečkou RFID bude koncipovaná tak, aby byla kompatibilní se současným systémem.

1. Seznamte se s ESP32 a prostředím ESP-IDF, stručně popiště základní vlastnosti tohoto procesoru.
2. Stručně popište dodaný hardware a jeho komunikační rozhraní.
3. Vytvořte program pro modul základny, který po připojení modulu do sítě LAN získá adresu z DHCP serveru a poté se bude pokoušet připojit k nadřazeným serverům. Každá z karet RFID i NFC bude mít svůj samostatný server.
4. Pro modul základny i čteček vytvořte programy, které budou spolu komunikovat prostřednictvím rozhraní RS422. Navrhněte takový komunikační protokol, kterým bude možné rozlišit komunikaci patřící čtečce RFID a čtečce NFC.
5. Pro modul čteček vytvořte program, který bude data z rozhraní RS422 přeposílat na jednotlivé čtečky. Identicky vytvořte program pro modul základny, který bude přeposílat data z rozhraní ethernet na rozhraní RS422.
6. Vytvořte program pro OS Linux, který bude naslouchat na daném serveru a bude přijímat spojení od čteček. Program bude poté komunikovat se čtečnou NFC a umožní vyčtení ID z přiložené karty.

Seznam doporučené odborné literatury:

[1] ESP32 Hardware Design Guidelines: ETSI, 37 s. Dostupné z WWW: <https://www.espressif.com/sites/default/files/documentation/esp32_hardware_design_guidelines_en.pdf>.
[2] Technical Specifications [online]. NFC Forum [cit. 2011-12-11]. Dostupné online https://nfc-forum.org
[3] Free RTOS Reference Manual, Dostupné z WWW: https://www.freertos.org/Documentation/FreeRTOS_Reference_Manual_V9.0.0.pdf
[4] BARR, Michael. Programming embedded systems in C and C++. Sebastopol, Calif.: O'Reilly, c1999. ISBN 1565923545.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. David Seidl, Ph.D.**

Datum zadání: 01.09.2018

Datum odevzdání: 30.04.2019

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 30. dubna 2019 .....

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Davidu Seidlovi, Ph.D. za konzultace a cenné rady při návrhu softwaru, dále také za rady při psaní bakalářské práce.

Abstrakt

Tato bakalářská práce se zabývá implementací nové části SW pro přístupový systém na VŠB – Technické univerzitě Ostrava, FEI. Nový přístupový systém podporuje technologii RFID a bezpečnější technologii NFC. Implementuje také použití E-Ink displeje. Je vytvořen nový komunikační protokol pro komunikaci dvou modulů v systému. Systém podporuje autokonfiguraci pomocí protokolu DHCP.

Klíčová slova: ESP32, NFC, RFID, C, Esp-Idf, FreeRTOS, přístupový systém

Abstract

This bachelor thesis is about implementation of a new part of a software for an access system on VŠB – Technická univerzita Ostrava, FEI. The new access system supports an RFID technology and a safer NFC technology. The thesis implements usage of an E-Ink display. New communication protocol was created for communication of two modules in the access system. The system supports autoconfiguration via DHCP protocol.

Key Words: ESP32, NFC, RFID, C, Esp-Idf, FreeRTOS, Access system

Obsah

Seznam použitých zkratek a symbolů

ACK	–	Acknowledgement, potvrzení příjmu
APDU	–	Application Protocol Data Unit, komunikační protokol mezi čtečkou karet a čipovou kartou
API	–	Application Programming Interface, rozhraní pro programování aplikací
ASCII	–	American Standard Code for Information Interchange, kódová tabulka, která definuje znaky anglické abecedy a jiné používané znaky
AWS	–	Amazon Web Services
Bd	–	Baud, jednotka modulační rychlosti
Bluetooth	–	otevřený standard pro bezdrátovou komunikaci
CAN BUS	–	Controller Area Network je sběrnice, využívaná nejčastěji pro vnitřní komunikační síť senzorů
CLK	–	Clock, hodinový signál
CMake	–	svobodný software pro automatizaci překladu programu
CRC	–	Cyclic redundancy check, cyklický redundantní součet
CS	–	Chip Select, vodič pro výběr periferie u SPI, stejné jako SS
DHCP	–	Dynamic Host Configuration Protocol, automatická konfigurace počítačů připojených do počítačové sítě
DHCP	–	Dynamic Host Configuration Protocol, název síťového protokolu
DMA	–	Direct Memory Access, přímý přístup do paměti
E-Ink displej	–	typ zobrazovací jednotky
FIFO	–	první dovnitř, první ven, obsluhovány v pořadí, v jakém do systému vstoupily
GIT	–	distribuovaný systém správy verzí
GNU	–	projekt zaměřený na svobodný software
GPIO	–	General-Purpose Input/Output, všeobecně použitelný vstupně výstupní pin
HDLC	–	High-Level Data Link Control, vysokoúrovňové řízení datového spoje, komunikační protokol
HW	–	Hardware, veškeré fyzicky existující technické vybavení
I^2C	–	multi-masterová počítačová sériová sběrnice
IDE	–	Integrated Development Environment, vývojové prostředí
IETF	–	Internet Engineering Task Force, organizace, která vyvíjí a podporuje internetové standard
IOT	–	Internet Of Things, Internet věcí

IP	–	Internet Protocol, protokol pracující na síťové vrstvě v počítačových sítích a internetu
IP adresa	–	jednoznačně identifikuje síťové rozhraní v počítačové síti, která používá IP protokol
IRQ	–	Interrupt, signál přerušení
JSON	–	JavaScript Object Notation, objektový zápis v jazyce javascript, způsob zápisu dat
KIT	–	souprava dílů/stavebnice, komplet
LAN	–	Local Area Network, počítačová síť
LCD	–	Liquid Crystal Display, displej z tekutých krystalů
LED	–	Light-Emitting Diode, světelná dioda
LEDC	–	Esp-Idf PWM modul primárně pro ovládání intenzity jasu LED diod
MAC	–	Media Access Control, jednoznačný identifikátor síťového zařízení
MCPWM	–	Motor Control PWM, PWM v Esp-Idf pro ovládání motorů a jiných zařízení
MIPS	–	jednotka výkonu procesoru, milion instrukcí za sekundu
MISO	–	Master In, Slave Out, Master Vstup, Slave výstup, název pinu u SPI
MOSI	–	Master Out, Slave In, Master výstup, Slave vstup, název pinu u SPI
MSYS2	–	kompilovací platforma pro MS Windows založená na Cygwin
NACK	–	Negative Acknowledgement, negativní potvrzení příjmu
NFC	–	Near Field Communication, technologie radiové bezdrátové komunikace
PPP	–	Point-to-Point protokol, komunikační protokol datové vrstvy
PSRAM	–	pseudostatická paměť
PWM	–	Pulse Width Modulation, pulzně šířková modulace
RAII	–	Resource Acquisition Is Initialization, zdroje inicializuje třída a spolu s třídou je zdroj uvolněn, programovací idiom používaný převážně v jazyce C++
RFID	–	Radio Frequency Identification, identifikace na radiové frekvenci
RISC	–	procesory s redukovanou instrukční sadou
ROM	–	Read Only Memory, nevolatilní paměť pouze pro čtení
RS232	–	způsob komunikace na sériové lince
RS422	–	sériový způsob komunikace
RS485	–	standard sériové komunikace
SAM	–	Secure Access Mode, používá se pro zabezpečený přístup
SMP	–	Symmetric Multiprocessing, druh víceprocesorových systémů s rovnocennými procesory

SPI	–	Serial Peripheral Interface, sériové periferní rozhraní
SS	–	Slave Select, vodič pro výběr periferie v SPI, stejné jako CS
STM	–	STMicroelectronics, výrobce polovodičů a elektroniky
SW	–	Software, programové vybavení
TCP/IP	–	Transmission Control Protocol/Internet Protocol, primární přenosový protokol/protokol síťové vrstvy
TFI	–	specifický identifikátor rámce pro PN532
TFTP	–	Trivial File Transfer Protocol, protokol pro přenos souborů
UART	–	Universal Asynchronous Receiver-Transmitter, univerzální asynchronní sériové rozhraní
UDP	–	User Datagram Protocol, přenosový protokol/protokol síťové vrstvy, bez záruky doručení
USB	–	Universal Serial Bus, univerzální sériová sběrnice
UTF-8	–	UCS/Unicode Transformation Format, způsob kódování znaků
V	–	Volt, jednotka napětí
WiFi	–	označení pro několik standardů IEEE 802.11 popisujících bezdrátovou komunikaci v počítačových sítích

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Tato bakalářská práce se zabývá vývojem jedné části software pro nově instalovaný přístupový systém na Fakultě elektrotechniky a informatiky VŠB – Technické univerzitě Ostrava. Hlavní motivací je nahradit původní systém přístupových terminálů založený na zastaralé technologii RFID novými terminály, které podporují použití nové a bezpečnější technologie NFC. Pro zajištění řešení nového přístupového systému byl na fakultě vyvinut HW, jehož základ je tvořen mikrokontrolérem ESP32 a komunikátorem PN532. Nové přístupové terminály nadále zůstávají kompatibilní s původním přístupovým systémem. Toto řešení umožňuje plynule přejít mezi těmito dvěma systémy.

Hlavním tématem této práce je vývoj aplikačního software pro zajištění přenosu dat mezi přístupovými terminály a řídícím přístupovým serverem. Přístupové terminály se skládají z modulu základny a čtečky, které jsou vzájemně propojeny sběrnicí RS422. Dále práce popisuje použití čtečky pro technologii RFID a použití nové čtečky karet podporujících protokol NFC. K nové čtečce podporující technologie NFC byla vytvořena knihovna pro její ovládání a přenos dat. Byl vytvořen software zajišťující otevírání elektronického zámku v rámu dveří na základě pokynu z nadřazeného serveru. Interakci s uživatelem zajišťuje barevné podsvícení displeje pomocí tříbarevných LED diod. Akustická signalizace je zajištěna pomocí piezoměniče.

V další části této práce je popsán vývoj software pro zajištění komunikace mezi modulem základny přístupového terminálu a modulem čtečky karet s E-Ink displejem. Dále byla řešena problematika možnosti načtení konfigurace přístupového terminálu pomocí konfiguračního souboru umístěného na síťovém serveru. Přístupový terminál je informován o umístění svého konfiguračního souboru prostřednictvím protokolu DHCP.

Aplikační software pro nový přístupový systém byl vyvinut na platformě Esp-Idf v jazyce C s využitím možností víceúlohového programování. Pro běh aplikací je využíván operační systém reálného času FreeRTOS.

Práce je rozdělena na dvě části, kde v první části je zpracován teoretický základ, který je pak v druhé části využit při popisu tvorby softwaru pro oba moduly přístupového terminálu.

2 Současný stav přístupového systému

V současnosti jsou k řízení přístupu do prostor fakulty používány RFID karty typu EM4100 pracující na frekvenci 125 kHz. Jako základna slouží převodník z RS232 na Ethernet. Základna v sobě nemá implementovanou žádnou logiku, pouze přeposílá data ze serveru na sériové rozhraní. Čtečka obsahuje mikroprocesor Atmel Tinny 2313, který je oproti novému řešení o mnoho méně výkonný. Součástí některých verzí stávající čtečky je dvouřádkový textový LCD displej. Stávající systém funguje jako přijímač jednoduchých příkazů ze serveru, jako je například: otevři zámek, zahraj melodii a podobně. Data odesílaná na server jsou pouze identifikační klíče karet (výrobní číslo), které jsou čtečkou načteny.

2.1 Výhody hadwarového návrhu nového systému

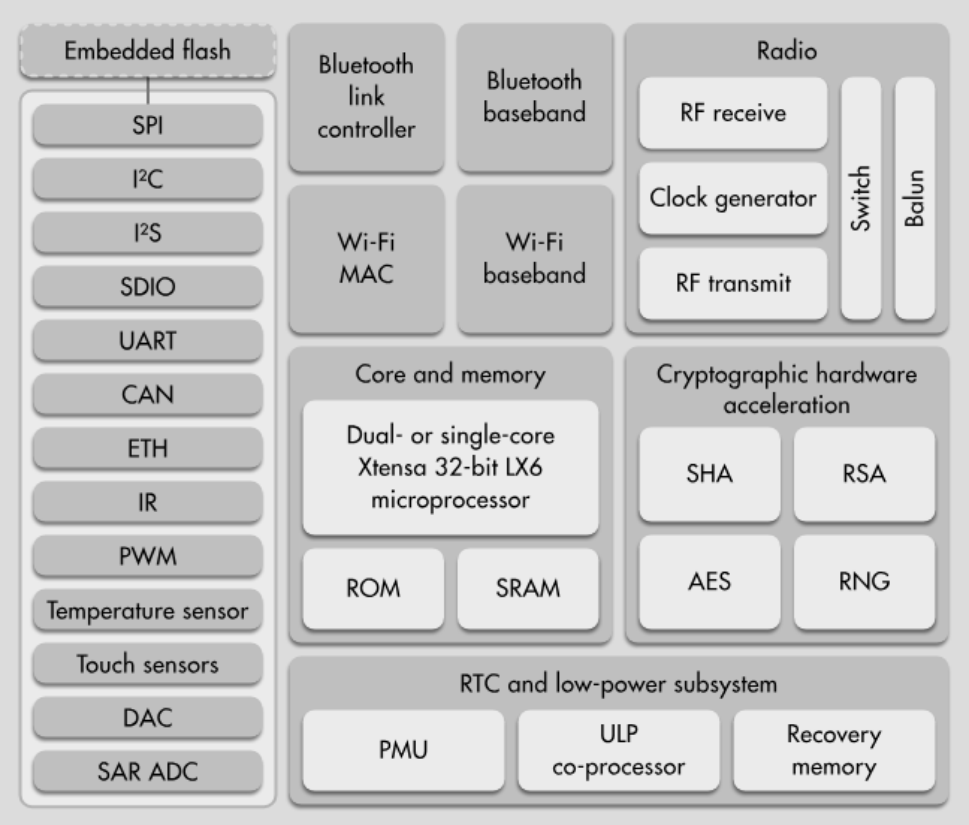
Oproti současně používanému systému má nově navržený hardware základny a čtečky mnoho vylepšených a nových vlastností. Především je použit značně výkonnější procesor, který umožňuje větší variabilitu při tvorbě software. Stávající LCD displej byl nahrazen moderní technologií E-Ink displeje, který je větší a čitelnější než původní. Hlavní změnou je především přítomnost čtečky pro technologii NFC, která je násobně bezpečnější a lze tuto technologii využít i pro pokročilejší funkce než pouhé ověřování identity uživatele.

2.2 Popis mikrokontroleru ESP32

ESP32 je cenově dostupný mikrokontroler s nízkou spotřebou, který je připraven pro komunikaci v síti technologie internetu věcí (IOT). ESP32 je nástupcem ESP8266, který byl levný WiFi mikročip vyrobený v roce 2014, obsahující TCP/IP stack. Od roku 2016 je produkován ESP32 firmou Espressif Systems se sídlem v Číně, která má jednu z poboček i v České republice. V ESP32 je osazen 32bitový dvoujádrový procesor. Dále obsahuje možnost konektivity prostřednictvím technologií Bluetooth a WiFi. Modul ESP32 má k dispozici 34 vstupně výstupních pinů, jejichž použití lze programově měnit. [1]

Podporuje velké množství rozhraní:

- I^2C
- SPI
- UART
- PWM
- CAN BUS
- SD karta
- Ethernet
- A další...



Obrázek 1: Funkcionality ESP32 [1]

2.3 Procesor použitý v modulu ESP32

Modul ESP32 je osazen 32bitovým symetrickým dvoujádrovým procesorem LX6 Xtensa od firmy Tensilica. Tento procesor je vyroben v technologii 40 nm a je schopen zpracovat až 600 milionů instrukcí za sekundu (MIPS). Je to little–endian procesor typu RISC podporující výpočty s plovoucí desetinnou čárkou (pouze však datový typ float, datový typ double je implementován softwarově), s nastavitelnou taktovací frekvencí, a to od 60 MHz až do 240 MHz. Procesor je Harvardského typu (oddělená paměť instrukcí a dat).

ESP32 podporuje přímý přístup do paměti (DMA) a to pro 13 periferií. Dále umožňuje také využití až 16MB externí flash paměti a připojit až 16MB PSRAM.

V této práci byl využit typ modulu ESP32-WROOM-32. V použitém modulu je osazena flash paměť o velikosti 4 MB. [2]

3 Vývojové frameworky – aplikační rámce

Oficiálně výrobcem podporované vývojové frameworky jsou Esp-Idf a Arduino, které jsou přístupné v programovacích jazycích C nebo C++. Díky podpoře komunity uživatelů na internetu pro tyto mikrokontrolery existují další možnosti použití pro různé aplikace. Některými z nich jsou Mongoose (IOT – primárně pro AWS), MicroPython nebo Espruino (používá programovací jazyk javascript). [3]

3.1 Arduino

Arduino je rozšířené a známé otevřené vývojové prostředí (původně pro mikrokontrolery ATmega od firmy Atmel). První vývojový KIT s názvem Arduino byl vyvinut v Itálii firmou Arduino v roce 2003. Vývojové prostředí pro tento projekt je často využíváno domácími uživateli především pro svou jednoduchost a nenáročnost. Vývojové prostředí Arduino obsahuje zjednodušené funkce pro využívání periferií, které se velice snadno používají. Díky značné oblíbenosti má mnoho volně šiřitelných knihoven. Programy napsané v tomto prostředí jsou složeny z funkce setup(), která proběhne při startu a funkce loop(), která se opakovaně volá. Tato typická vlastnost Arduina byla implementována především pro snadné pochopení funkce programu i pro nezkušené a začínající programátory. [4]

3.2 Esp-Idf

Esp-Idf je framework vyvíjený přímo výrobcem modulu ESP32 firmou Espressif Systems. Od roku 2018 je dostupný ve verzi 3, která je částečně kompatibilní se starší verzí 2. Je to otevřený framework založený na FreeRTOS a obsahuje bohatou knihovnu ESP32 specifických funkcí. Dále framework obsahuje TCP/IP stack lwIP. lwIP je otevřený TCP/IP stack, který si klade důraz na nízké využití zdrojů, kterých ve vestavěných systémech nebývá mnoho. [5]

Jednou ze zajímavých možností Esp-Idf je, že lze použít knihovny napsané pro prostředí Arduino, s tím, že se musí Arduino přidat jako komponenta. To sice zvýší nároky na velikost binárního souboru, ale výhodou je možnost využití již napsaných knihoven.[3]

Sestavovací systém pro Esp-Idf využívá programu make a skriptů v jazyce Python. V budoucích verzích (verze 4 a výše) se chystá využít CMake a to z důvodu lepší podpory v různých vývojových prostředích (dále IDE). Sestavovací systém také obsahuje konfigurační program, který se spouští příkazem „make menuconfig“. Konfigurační program umožňuje nastavit pokročilé konfigurace celého systému. Mezi další součásti sestavovacího systému patří také volba „monitor“, která umožňuje sledování modulu při běhu programu.

V současnosti je oficiálně doporučováno použít jako vývojové prostředí program Eclipse.[6] Lze ale také využít IDE PlatformIO postavené na Visual Code.

3.2.1 Možnosti nastavení Esp-Idf

Pro konfiguraci Esp-Idf se používá program make menuconfig, který má podobný vzhled a použití jako při nastavování kompilace kernelu operačního systému Linux. Umožňuje konfigurovat nahrání nového firmware do ESP32. (baud rate, port, . . .). Dalšími užitečnými nastaveními jsou:

Stack smashing protection - ochrana přetečení zásobníku

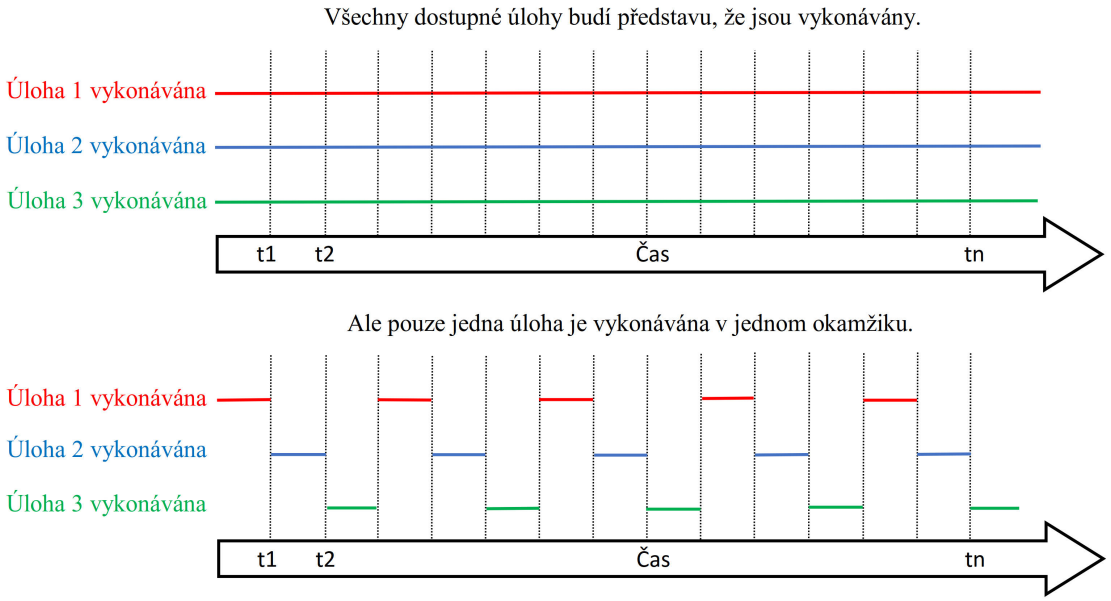
Heap corruption detection - ve třech možných nastaveních umožňuje detekovat chyby práce s pamětí

Partition table - nastavení velikosti a počtu oddílů

3.3 FreeRTOS

Je operační systém reálného času (RTOS) vyvinutý firmou Real Time Engineers Ltd.. FreeRTOS dává záruku o dokončení činností v určitém čase. Je vytvořen se zaměřením na vestavěné systémy, takže je kladen důraz na nízkou paměťovou náročnost a nízkou režii. Umožňuje vytvořit abstrakci nad časovými událostmi.

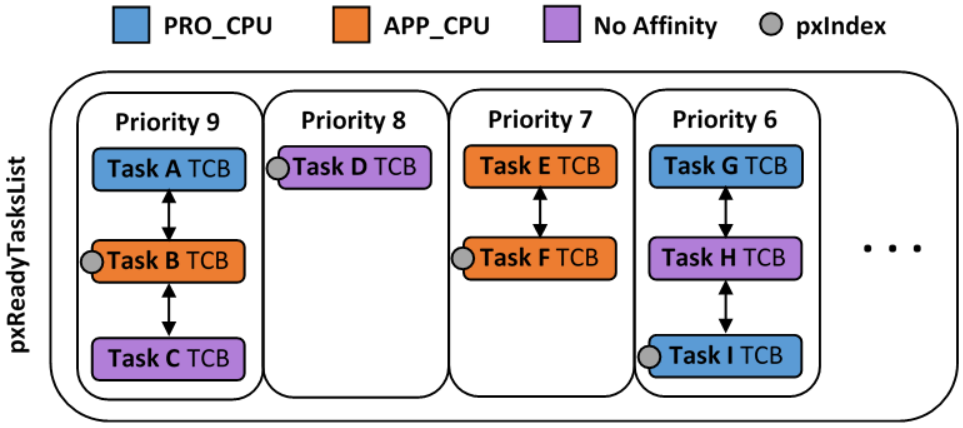
Aplikace ve FreeRTOS jsou složené z nezávislých úloh (task). Časovač (real time scheduler), pak určuje která úloha se má vykonávat a přepíná mezi nimi (zastavuje je a spouští). Každá úloha má svůj vlastní zásobník (stack), který časovač ukládá a obnovuje – context switching. FreeRTOS umožňuje tedy multitasking. V klasickém FreeRTOS se vykonává v jedné chvíli pouze jedna úloha. [7]



Obrázek 2: Více úloh na jednom jádře [7]

3.4 Úpravy FreeRTOS v Esp-Idf

Jelikož ESP32 má dvě jádra, tak je FreeRTOS upraven pro optimální využití této vlastnosti. Protože jsou v ESP32 jádra identická a sdílí stejnou paměť, úlohy můžou být spouštěny na jakémkoliv z jader. Funguje tedy symetrický multiprocesing (SMP). Využívá round robin časovač, který však ignoruje více jak dvě úlohy v čekajícím stavu se stejnou prioritou, proto se musí úlohy rozprostřít do více priorit. Přítomnost multitaskingu a více jader sebou nese nutnost řešit možný souběh v programu, například pomocí semaforů a kritických sekcí. [7]



Obrázek 3: Ilustrace datové struktury seznamu připravených úloh [8]

FreeRTOS nabízí taky prostředky pro práci ve víceúlohovém systému, které z důvodu dvou jader v ESP32 mají o to větší důležitost. FreeRTOS nabízí API pro fronty (queue), semafory, softwarové časovače a události (event groups). V této práci jsem využil většinu z těchto funkcionalit.

3.5 Prostředky nabízené FreeRTOS

- Semafor (Semaphore)
  - Ve FreeRTOS se dělí na binární a počítající (counting)
  - Implementovány pomocí front
- Fronta (Queue)
  - Umožňují specifikovat dobu blokování
  - FIFO – první dovnitř, první ven
  - Lze dávat prvky i na začátek fronty
- Časovač (Software Timer)
  - Umožňuje spustit funkci za daný časový interval

  - Nespotřebovává žádný strojový čas, dokud nevyprší
  - Volaná funkce nesmí blokovat
- Události (Event bits)
  - Indikují, zda nastala událost
  - Dokumentace FreeRTOS doporučuje zvážit možnost jejich použití jako lehčích binárních semaforů
  - Efektivnější než semafor (o 45 %)

FreeRTOS nabízí také pět typů alokace paměti [8]:

1. Nejjednodušší, neumožňuje uvolnění alokované paměti
2. Umožňuje uvolnění alokované paměti, ale nespojuje sousední volné bloky paměti
3. Využívá standardních free a malloc funkcí, ale obaluje je, aby byly thread safe
4. Spojuje sousední bloky paměti, aby zabránilo fragmentaci paměti
5. Umožňuje, aby halda byla rozprostřena na více nesousedních blocích paměti

Esp-Idf oficiálně podporuje použít pouze pátý typ. [5]

3.6 Výběr mezi prostředím Arduino a Esp-Idf

Prostředí Arduino se sice jeví na první pohled jako jednodušší volba díky jednoduchému programovacímu rozhraní, ale má řadu nedostatků. Nemá tak rozsáhlé možnosti a některé pokročilejší funkce (např. práce s přerušeními a piny nebo se sběrnicí SPI) jsou značně zjednodušené.

V případě použití prostředí Arduino pro programování mikrokontroleru ESP32 jsou tyto skutečnosti silně limitující, protože ESP32 má bohatou nabídku periferií a možností jejich nastavení. V prostředí Arduino je také komplikovaná práce se síťovými sockety. Esp-Idf je z tohoto pohledu daleko lépe připraven na aplikace využívající pokročilejší možnosti všech dostupných periferií.

Příklad rozdílné složitosti lze ilustrovat na rozdílech těchto dvou zdrojových kódů, kde kód pro Arduino provede o mnoho více, ale s menší možností konfigurovatelnosti.
Kód pro inicializaci rozhraní SPI a jeho použití:

```
hspi = new SPIClass(HSPI);
vspi->begin();
vspi->beginTransaction(SPISettings(spiClk, MSBFIRST, SPI_MODE0));
digitalWrite(SS, LOW);
vspi->transfer(data);
digitalWrite(SS, HIGH);
```

```
vspi->endTransaction();
```

Výpis 1: Konfigurace a poslání jedné transakce v Arduino

Pouze odeslání dat pomocí rozhraní SPI v Esp-Idf:

```
trans_desc.length = 8;
trans_desc.rxlength = 0;
trans_desc.user = (void *) 0;
trans_desc.tx_buffer = &command;
trans_desc.rx_buffer = NULL;
ESP_ERROR_CHECK(spi_device_transmit(epd->handle, &trans_desc));
```

Výpis 2: Poslání jedné transakce v Esp-Idf

4 Stručný popis hardware nového přístupového systému

V práci budou využity dva oddělené moduly, které budou spolu komunikovat pomocí sběrnice RS422. Jeden modul je určen pro interakci s uživatelem a čtení karet (čtečka). Druhý modul pro komunikaci se síťovými servery prostřednictvím sítě LAN (základna).

4.1 Modul základny

Modul základny obsahuje a využívá tyto technologie:

- Ethernet
- Převodník RS422
- Piezoměnič
- Spínací relé
- USB rozhraní pro programování

4.2 Modul čtečky

Modul čtečky obsahuje a využívá tyto technologie:

- Převodník RS422
- Piezoměnič
- USB rozhraní pro programování
- RFID čtečku RMD6300
- NFC čtečku PN532
- E-Ink displej
- LED diody pro nasvícení displeje

4.3 Asynchronní sériová komunikace

Komunikace mezi zařízeními je založena na vysílání jednotlivých bitů dat, které je zapotřebí přenést mezi moduly. Tento proud dat je doplněn dalšími bity sloužícími k synchronizaci a detekci chyb. Universální asynchronní přijímač a vysílač (UART) nepoužívá synchronizaci na úrovni bitů, ale na úrovni větších celků (typicky 8 bitů, ale volitelně i 7, 6 nebo 5 bitů). [9]

Přijímač i vysílač obou zařízení musí být předem nastaven. Pro bezchybnou komunikaci musí být nastavena na obou stranách komunikace stejná přenosová rychlost v bitech (bitrate), pak také počet přenášených bitů, který bude přenášen v jednom celku, délka stop bitu a případně i další doplňující bity. [10]

Před odesláním dat se přenáší jako první start bit, tzn. že logická úroveň na přenosové lince se změní z logické jedničky na logickou nulu. Na začátku přenosu dat se vždy změní úroveň klidového

stavu, což je v tomto případě logická jednička. Přijímač detekuje tento start bit a na základě informací o přenosové rychlosti a počtu přenášených bitů přijme data. Na konci přenosu dat se přenáší stop bit, který má úroveň logické jedničky, a jeho délka je konfigurovatelná. Díky stop a start bitu se může přijímač synchronizovat s vysílačem. Správná detekce start bitu je u asynchronního přenosu velmi důležitá, protože od tohoto bitu jsou odvozeny časy přenosu ostatních bitů.

Při komunikaci lze také využít paritního bitu, který umožňuje detekovat chyby. Parita může být lichá nebo sudá a určuje se podle počtu logických jedniček. Systém parity neumí zachytit všechny chyby a především neumožňuje zjistit, který bit je chybně. Proto je vhodnější využit systém detekce chyb, který pracuje nad většími celky. Jde například o Hammingovo kódování nebo cyklický redundantní součet (CRC). [11]

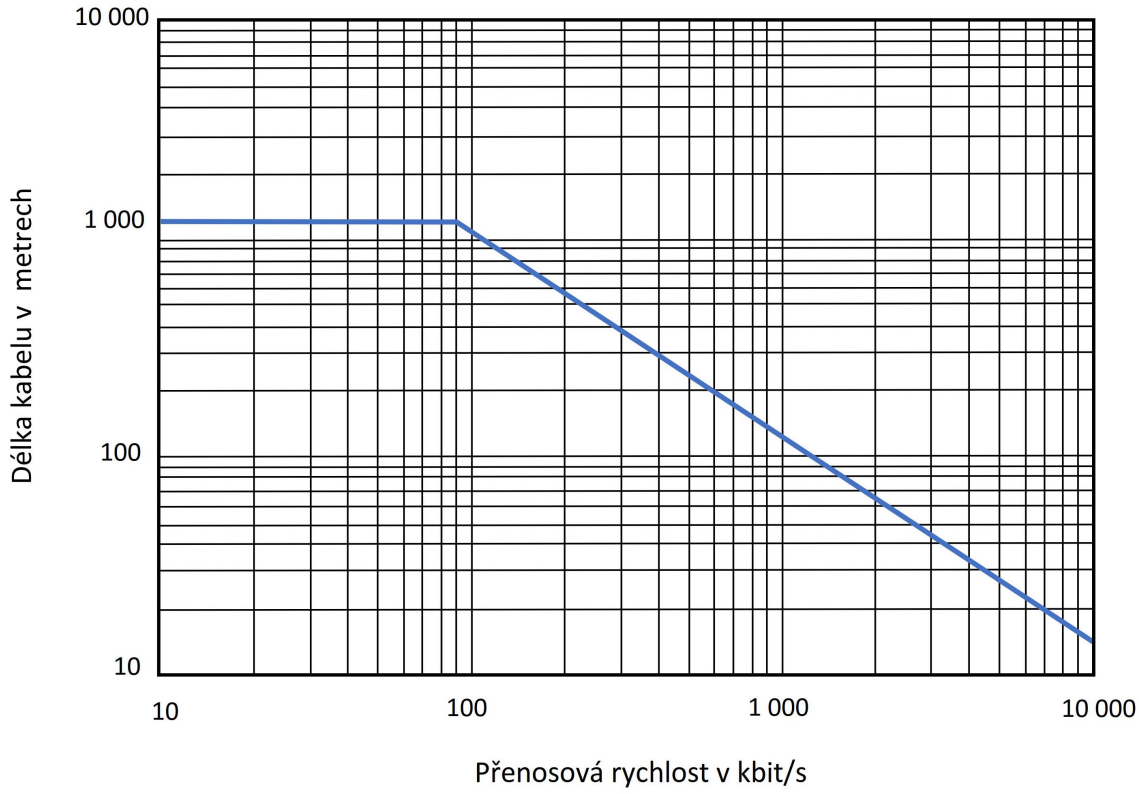
Konvenční zápis konfigurace UART se zapisuje ve formátu XXXX/D-P-S, kde XXXX je rychlost přenosu, D počet datových bitů, P typ paritního bitu (N – není použit, E – lichý, O – sudý) a S je délka stop bitu. Např. 9600/8-N-1 značí rychlost 9 600 bit/s, osm datových bitů, žádný paritní a délka stop bitu je jedna.

4.4 Rozhraní RS232

Je rozhraní typu universální asynchronní přijímač a vysílač bez nutnosti modulace z roku 1960. Obsahuje definice napěťových úrovní, časování a dalších elektrických signálních charakteristik. Dále také rozhraní, identifikaci pinů, funkce jednotlivých obvodů. Napěťové úrovně se pohybují od ± 3 V až do ± 15 V. U RS232 se většinou používají přenosové rychlosti odvozené od hodnoty 115 200 Bd. Tedy o řadu 115 200, 57 600, 38 400, 28 800, 23 040, 19 200, 9 600, 4 800, 2 400. U rozhraní RS232 odpovídá jeden baud 1 bitu za sekundu (ne však rychlosti datového přenosu, protože některé bity jsou využity pro start, stop a případně i paritní bity.) [12]

4.5 Rozhraní RS422

RS422 je standard pro sériovou komunikaci využívající diferenciální signál, byl specifikovaný v roce 1996. Při přenosu dat se využívá rozdílu potenciálu mezi vodiči. Tento standard umožňuje použít přenosovou rychlost až 10 Mb/s. Komunikovat lze až do vzdálenosti 1500 metrů. Napěťové úrovně jsou -6 V a $+6$ V. Se zvyšující délkou kabelu se snižuje maximální přenosová rychlost, které lze dosáhnout (viz. Obrázek 4). Oproti standardu RS485 je spojení pouze dvoubodové. Proto jsou pro komunikaci modulů základny a čtečky použity dva páry vodičů, umožňující obousměrnou komunikaci. Na úrovni logických signálů je komunikace stejná jak u rozhraní RS232. [13]



Obrázek 4: Závislost přenosové rychlosti na vzdálenosti při použití RS422 [13]

4.6 UART v ESP32

Pro použití UART v ESP32 je prvně zapotřebí vytvořit konfiguraci, která určuje základní nastavení UART, jako je počet datových bitů, parita a další. Tato konfigurace se poté nastaví na určitý ESP32 UART kanál. Na daném UART kanálu se pak určí, které piny se mají nastavit, a poté se tento kanál musí inicializovat.

```
const uart_config_t uart_config = { .baud_rate = 115200, .data_bits =
    UART_DATA_8_BITS, .parity =
        UART_PARITY_DISABLE, .stop_bits = UART_STOP_BITS_1, .flow_ctrl =
            UART_HW_FLOWCTRL_DISABLE };
   uart_param_config(UART_NUM_1, &uart_config);
   uart_set_pin(UART_NUM_1, TXD_PIN, RXD_PIN, UART_PIN_NO_CHANGE,
      UART_PIN_NO_CHANGE);
   uart_driver_install(UART_NUM_1, RX_BUF_SIZE * 2, 0, 0, NULL, 0);
}
```

Výpis 3: Konfigurace UART

Poté už lze jednoduše číst nebo zapisovat na UART.

4.7 Rozhraní SPI

Jedná se o rozhraní pro synchronní sériovou komunikaci na krátké vzdálenosti. SPI má samostatný vodič pro hodinový signál (CLK), je plně duplexní a má dva vodiče pro přenos dat (MISO, MOSI). V případě nutnosti komunikace s více periferiemi je použit vodič sloužící pro výběr periferie, se kterou se komunikuje (SS – slave select nebo CS – chip select). Na sběrnici SPI je jeden master, který řídí komunikaci pomocí hodinového signálu, a určuje se, s kterým zařízením na sběrnici bude komunikovat. Další zařízení jsou v módu slave a komunikují podle hodinového signálu pouze, pokud jsou adresovány pomocí SS. [14]

V ESP32 se musí nastavit sběrnice SPI, kde se definují, jaké piny se mají použít, a ostatní parametry, jako jsou vlajky (například vlajka pro nejvíce signifikantní bit) a maximální délka dat k přenosu nebo callback funkce.

```
const uart_config_t uart_config = {
spi_bus_config_t buscfg = { .miso_io_num = PIN_NUM_MISO, .mosi_io_num =
   PIN_NUM_MOSI, .sclk_io_num = PIN_NUM_CLK, .quadwp_io_num = -1, .
       quadhd_io_num = -1, .max_transfer_sz = MAX_TRANSFER_SZ };
}
```

Výpis 4: Konfigurace SPI sběrnice

Poté se přidá SPI zařízení, kde se nastavuje slave select pin a specifické nastavení pro dané zařízení. Na jedné SPI sběrnici může být více zařízení. Samotná data se přenášejí pomocí SPI transakcí, které zajišťují atomicitu přenosu. U transakce se musí nastavit délka, buffer (vyrovnávací paměť) a další vlastnosti nutné k přenosu. V Esp-Idf je možné využít DMA (direct memory access).

```
spi_transaction_t t{};
memset(&t, 0, sizeof(t));
t.length = 8;
t.cmd = 0x02;
t.rxlength = 8;
t.rx_buffer = (void *) &status_reading_response;
t.tx_buffer = NULL;
esp_err_t ret = spi_device_transmit(spi, &t);
```

Výpis 5: Použití transakce

4.7.1 Přímý přístup do paměti – Direct memory access

ESP32 umožňuje pro komunikaci s rozhraním SPI použít přímý přístup do paměti (DMA). Umožňuje zařízení připojenému přes SPI zapisovat přímo do paměti ESP32 bez nutnosti obsluhy procesorem. Paměť, kterou lze takto použít, musí být speciálně alokovaná se zvláštním příznakem

pro DMA, aby alokovaná data byla umístěna do části paměti, která DMA podporuje.
Kód pro alokaci paměti vhodné pro DMA:

```
heap_caps_malloc((epd->width * epd->height / 8), MALLOC_CAP_DMA);
```

Výpis 6: Alokace paměti použitelné pro DMA

4.8 Technologie RFID

RFID neboli identifikace na rádiové frekvenci, je technologie umožňující identifikaci pomocí RFID štítků (tag). Technologie RFID byla definována v roce 1983 a mělo se jednat o náhradu identifikace pomocí čárových kódů. Štítky využívané v RFID obsahují elektronicky uložené informace. Specifikace umožňovala využívat štítky zapisovatelné (elektronická peněženka) nebo pouze ke čtení.

4.8.1 Princip

Obecně se technologie RFID dělí na aktivní a pasivní. Aktivní má svůj zdroj napájení a má dosah až desítky metrů. Pasivní nemá svůj zdroj napájení a místo toho je napájen elektromagnetickou energií ze čtečky. Dosah pasivních štítků je zhruba do 25 cm. Pasivní štítek využije vysílanou elektromagnetickou energii k nabití svého kondenzátoru a následně k odeslání odpovědi. Obvykle vyšle unikátní identifikační číslo zadané při výrobě štítku. [14]

Podle standardu ISO/IEC 18000 RFID operuje na frekvencích 125 kHz až 134 kHz a 13,56 MHz, v Evropě také na frekvenci 865–868 MHz a v Severní Americe 902–928 MHz. Každá z frekvencí má jiný dosah a použití.

Zařízení používající frekvence 125–134 kHz jsou více odolné proti rušení, ale mají menší čtecí dosah a pomalejší rychlost přenosu dat.

Frekvence 13,56 MHz je méně odolnější proti rušení, ale má větší dosah a vyšší rychlost přenosu dat. Může také mít menší anténu než RFID pracující na frekvencích 125–134 kHz. RFID technologii na frekvenci 13,56 MHz využívá a rozšiřuje technologie NFC. [15]

4.8.2 Čtecí modul RMD6300

Pro čtení RFID karet je použit cenově dostupný modul RMD6300, který je vyráběn v Číně. Umožňuje číst RFID karty pracující na frekvenci 125KHz typu EM4100. Karty typu EM4100 mají 64bitovou paměť a jsou určeny jen pro čtení. Modul podporuje komunikaci pomocí asynchronní sériové komunikace (UART) v RS232 formátu s napěťovou úrovní 3,3 V.

Tento modul komunikuje rychlostí 9 600 baud, kde je osm bitů dat, jeden stop bit a žádný paritní bit. Formát dat zasílaný čtečkou karet RFID je následující: na začátku byte 0x02 a na konci 0x03.

Po počátečním byte je deset bytů dat v ASCII formátu. Předposlední byte je kontrolní součet, který je exkluzivní disjunkce (operace XOR) dvojic bytů výrobního čísla karty. [17]

Tabulka 1: Formát dat zasílaných čtečkou RMD3600

0x02	10 ASCII znaků	Kontrolní součet	0x03

4.8.3 Nevýhody a bezpečnost současného přístupového systému

Karty typu EM4100, které jsou nyní použity na VŠB – Technické univerzitě Ostrava, lze velmi jednoduše zkopírovat, a to zařízením v ceně do dvou set korun. Tato zařízení jsou volně dostupná na internetu. [18][19]

Na základě výše uvedeného nelze stávající systém založený na tomto typu karet považovat za dostatečně bezpečný.

4.9 Technologie NFC

NFC rozšiřuje RFID na frekvenci 13,56 MHz. Technologie NFC je popsána standardem ISO/IEC 18092, který vychází ze standardu ISO/IEC 14443 (standard definuje bezkontaktní čipové karty pro identifikaci a jejich komunikaci) a je tedy s ISO/IEC 14443 částečně kompatibilní. ISO/IEC 18092 definuje jiný přenosový protokol a tím nahrazuje část 4 standardu s ISO/IEC 14443. [20]

4.9.1 Princip

NFC je sada technologií pro přenos dat na velmi krátkou vzdálenost s obvyklými přenosovými rychlostmi od 106 kbit/s do 424 kbit/s. Dělí se podobně jako RFID na pasivní a aktivní podle druhu napájení. Aktivní mají vlastní zdroj napájení a pasivní jsou napájeny aktivními zařízeními. NFC má tři operační režimy:

- Tag reader/writer – umožňuje číst a zapisovat z informace z tagů
- Peer to peer – umožňuje komunikaci mezi dvěma aktivními NFC zařízeními
- Card emulation – umožňuje se NFC zařízením chovat jako chytré bezkontaktní karty podle standardu ISO/IEC 14443

4.9.2 Formát příkazů – APDU

APDU je komunikační protokol mezi čipovou kartou a čtečkou definovaný normou ISO/IEC 7816-4. Skládá se ze dvou kategorií zpráv – APDU příkaz a APDU odpověď.

Příkaz APDU je posílán čtečkou čipové kartě a obsahuje 4 bytovou hlavičku a poté až 65 535 bytů dat. [21]

Tabulka 2: Příkaz APDU

CLA	INS	P1-P2	Lc	Data	Le

CLA – typ příkazu

INS – specifický příkaz

P1-P2 – parametry

Lc – délka dat

Le – maximální délka odpovědi

Odpověď APDU se skládá z dat odpovědi a dvou bytů indikujících stav provedení příkazu.

4.9.3 Komunikátor pro NFC – PN532

PN532 je transceiver (vysílač i přijímač) pro bezdrátovou komunikaci pracující na frekvenci 13,56 MHz, který vyrábí společnost NXP. Obvyklý dosah PN532 je 50 mm. V PN532 je použit mikrokontroler 80C51 od firmy Philips, který obsahuje 40 kB ROM paměti a 1 kB RAM paměti. PN532 obsahuje SAM (security access module) modul, který má zabezpečenou paměť (pro potřeby bezpečného uložení klíčů) a podporuje kryptografické funkce (šifrování, počítání el. podpisů, ověřování el. podpisů, apod.). SAM modul se využívá pro zvýšení zabezpečení systémů s bezkontaktními kartami. SAM modul také umožňuje vzájemnou autentizaci. PN532 podporuje tato hostitelská rozhraní [22]:

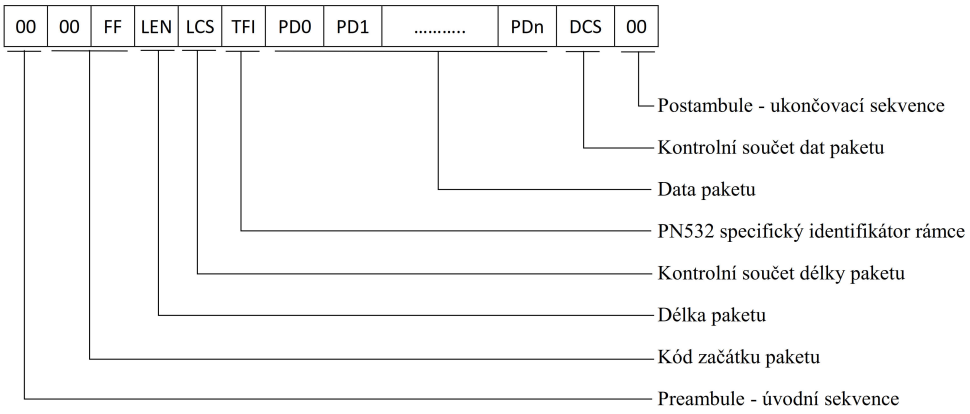
- I^2C
- SPI
- HSU (high speed UART – vysoko rychlostní UART)

Kromě těchto rozhraní je možné využít pin označený jako IRQ (přerušení), který informuje, zda je pro nadřazený systém připravená odpověď.

Komunikace mezi PN532 a ovládajícím zařízením je prováděna pomocí rámců v poloduplexním režimu. PN532 má tyto typy rámců:

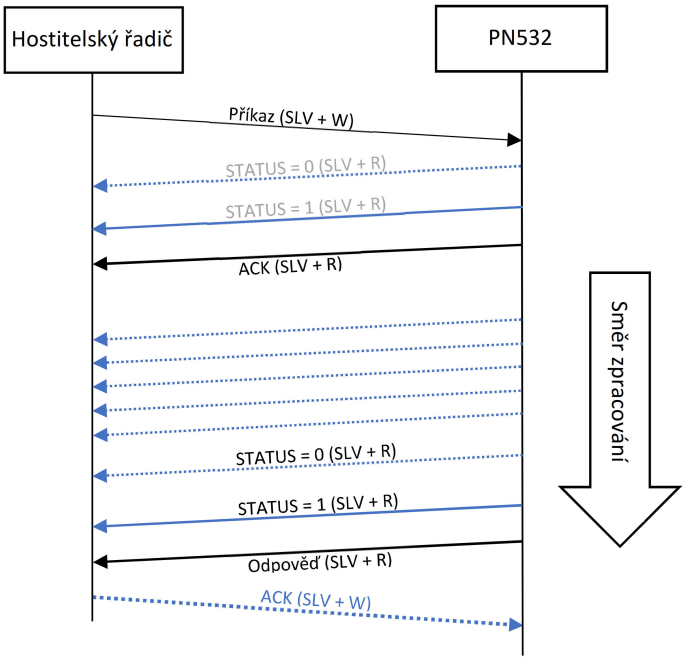
- normální informační rámec
- rozšířený informační rámec – pro rámce větší jak 255 byte
- potvrzovací (ACK) rámec – informuje o úspěšném přečtení zprávy
- negativně potvrzovací (NACK) rámec – hostitel používá pro znovu poslání odpovědi z PN532
- rámec o chybě (error frame) – informuje o chybě na aplikační úrovni

Normální informační rámec má strukturu:



Obrázek 5: Struktura normálního rámce [22]

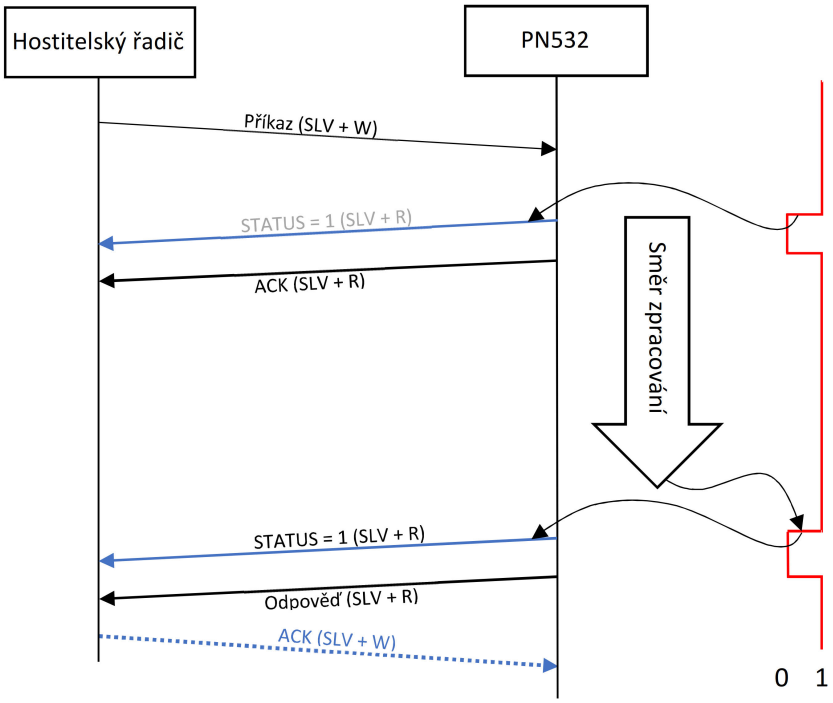
Struktura paketu se skládá ze začátku paketu, kterému předchází preambule složená z libovolného množství nenulových byte a jednoho nulového, následuje délka dat a kontrolní součet pro délku dat v rámci. TFI označuje směr rámce (z PN532 nebo do PN532). Následují data a po nich jednobytový kontrolní součet pro data (DCS). Na konci paketu je ukončovací sekvence.



Obrázek 6: Obecný princip komunikace [22]

V rozšířeném informačním rámci má délka rámce i kontrolní součet hodnotu 0xFF. Poté je délka určena z dvou následujících bytů, kde délka je první byte krát 256 plus druhý byte. Byte, které následují, jsou stejného významu jak u nerozšířeného informačního rámce.

Při komunikaci přes rozhraní SPI musí host poslat první byte, který informuje, o jakou operaci se bude jednat. Typy operací jsou čtení dat, zápis dat a přečtení stavu (indikuje, zda můžeme číst z PN532). Při čtení stavu odpovídá PN532 jedním bytem, kde poslední bit značí, zda je možné přečíst data z PN532, nebo je nutné opakovat dotaz. V případě použití handshake a IRQ pinu, tak nutnost čtení stavu odpadá, protože IRQ pin dá informaci o možnosti čtení dat z PN532. Není tedy nutné opakované testování stavu PN532. (viz. obrázek 7)



Obrázek 7: SPI komunikace s použitím IRQ pinu [22]

PN532 podporuje tyto režimy:

- ISO/IEC 14443A/MIFARE Reader/Writer
- FeliCa Reader/Writer
- ISO/IEC 14443B Reader/Writer
- ISO/IEC 14443A/MIFARE Card MIFARE Classic 1K or MIFARE Classic 4K card
- Emulation mode

- FeliCa Card emulation
- ISO/IEC 18092, ECMA 340 Peer-to-Peer

4.10 Spínací relé

Spínací relé se nachází na modulu základny a slouží k přivedení napětí na kontakty elektronického dveřního zámku. V případě přístupového systému se spínací relé využívá pro otevírání jednotlivých dveří. V rámu dveří se nachází elektrický dveřní zámek. Dveře je možné otevřít, pokud na kontaktech elektrického dveřního zámku je přítomno napětí. Relé se ovládá pomocí programovatelného vstupního pinu (GPIO).

Esp-Idf umožňuje pro GPIO nastavení pulldown a pullup rezistoru a různé typy přerušení. Poté se GPIO ovládají jednoduše:

```
gpio_set_level(GPIO_NUM_32, 0);
```

Výpis 7: Ovládání GPIO

4.11 Modulace PWM

K ovládání LED diod nasvícení čtečky a piezoměniče čtečky se využívá pulzně šířková modulace (PWM). PWM je diskrétní modulace pro přenos analogového signálu pomocí hodnot digitálního signálu.[23] ESP32 podporuje jeden hardwarový PWM kanál a 16 softwarově implementovaných PWM kanálů. Jsou podporovány dva typy PWM. Jeden typ je MCPWM (Motor control PWM) a druhý typ LEDC. MCPWM je určen k ovládání motorů. Typ LEDC je přímo určen k ovládání jasu LED diod. Jeho výhoda je především velká spínací frekvence (až 40MHz).

Ke generování PWM pro LED diody a ozvučení je využit modul LEDC, který je primárně určen pro kontrolu intenzity LED světel. Jde však využít i k jiným účelům. Jednotlivé kanály PWM se dělí na dva typy, pomalé a rychlé, LEDC má osm časovačů (timer) a 16 použitelných kanálů. LEDC umožňuje široké rozlišení pracovního cyklu PWM, ale se stoupající frekvencí je snižována možnost rozlišení střídy, a tudíž pro frekvenci 40 MHz je pouze rozlišení jeden bit (50% pracovní cyklus), zatímco pro 13 kHz je možné rozlišení až 13 bitů. To znamená, že pracovní cyklus může být nastaven od 0 % do 100 % s rozlišením 0,012 %. [24]

LEDC umožňuje také hardwarové slábnutí nebo zesilování (fade) PWM. To umožňuje provádět změny bez nutnosti obsluhy procesorem. Pro použití LEDC je prvně nutno nastavit časovač s danou rychlostí a rozlišením a frekvencí:

```
ledc_timer_config_t timer_conf;
timer_conf.speed_mode = LEDC_HIGH_SPEED_MODE;
timer_conf.duty_resolution = LEDC_TIMER_10_BIT;
timer_conf.timer_num = LEDC_TIMER_0;
timer_conf.freq_hz = 100;
```

```
ledc_timer_config(&timer_conf);
```

Výpis 8: Nastavení LEDC časovače

Poté se musí nastavit kanál, který bude využívat nastavený časovač. U kanálu musíme nastavit na kterém pinu se bude signál PWM generovat, jeho rychlostní režim, použitý časovač, jeho pracovní cyklus a zda má nastat přerušení, když je dokončeno slábnutí nebo zesilování:

```
ledc_channel_config_t blue_led;
blue_led.gpio_num = GPIO_NUM_33;
blue_led.speed_mode = LEDC_HIGH_SPEED_MODE;
blue_led.channel = LEDC_CHANNEL_0;
blue_led.hpoint = 0;
blue_led.intr_type = LEDC_INTR_DISABLE;
blue_led.timer_sel = LEDC_TIMER_0;
blue_led.duty = 0x0;
ledc_channel_config(&blue_led);
```

Výpis 9: Nastavení LEDC kanálu

Například ke změně intenzity LED diod lze pak využít funkci, která změní pracovní cyklus daného kanálu:

```
void set_duty_percentage(uint8_t percent, ledc_channel_t channel) {
    uint32_t max = 1 << 10;
    if (percent > 100)
        percent = 100;
    uint32_t value = max * percent / 100;
    if (value >= max)
        value = max - 1;
    ESP_ERROR_CHECK(ledc_set_duty(LEDC_HIGH_SPEED_MODE, channel, value));
    ESP_ERROR_CHECK(ledc_update_duty(LEDC_HIGH_SPEED_MODE, channel));
}
```

Výpis 10: Nastavení pracovního cyklu

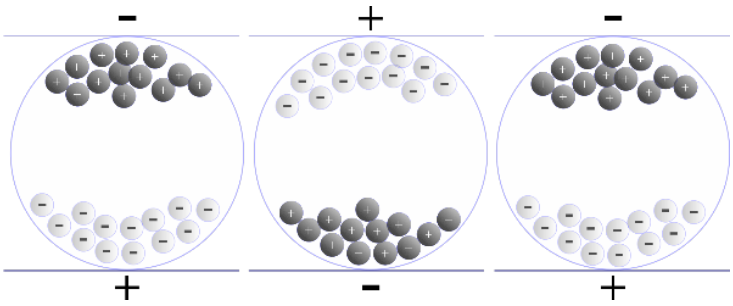
4.12 E-Ink

Elektronický papír (E-Ink) je typ zobrazovací jednotky, který napodobuje vzhled papíru a je čitelný i na přímém slunečním svitu. E-Ink byl vyvinut firmou E Ink Corporation. Nabízí také velmi dobré pozorovací úhly. E-Ink je schopen uchovávat zobrazovaný text nebo obrázek bez spotřeby elektrické energie.[25] Elektrická energie se spotřebovává pouze při překreslení a komunikaci. E-Ink displej je možné vyrobit také z ohebných materiálů. Jeho nevýhodou je velmi nízká

obnovovací frekvence (jednotky sekund) a omezený počet barev (většinou dvoubarevné až tříbarevné). [26]

4.12.1 Princip zobrazení

Malé uzavřené kapsle o velikosti desítek či stovek μm tvoří jednotlivé body obrazu. V kapslích se nachází průsvitný elektroforetický roztok. Tento roztok obsahuje záporně nabité černé částice (inkoust) a kladně nabité bílé částice. Částice mají řádově velikost jednotky μm. Kapsle jsou umístěny mezi elektrody a přivedením napětí na elektrody se částice podle svého náboje přesunou k elektrodě s opačnou polaritou. [27]



Obrázek 8: E-Ink kapsle [27]

4.12.2 Ovládání

Instalovaný E-Ink modul je 2,7palcový e-Paper HAT od společnosti Waveshare. Jeho rozlišení je 264 x 176 pixelů. K ovládání modulu se používá 4 pinové SPI rozhraní a také jeden pin pro reset displeje. Použitý modul E-Ink má obnovovací frekvenci 6 sekund. [28]

Pro ovládání E-Ink displeje je připravená struktura a funkce nad ní operující, které komunikují s E-Ink displejem a jsou inspirovány kódem dodávaným jako ukázkovým pro procesory STM. Pro vykreslování textu a obrazců je využita knihovna od společnosti Waveshare.

```
typedef struct {
   uint8_t * frame_buffer;
   gpio_num_t rst;
   gpio_num_t dc;
   gpio_num_t cs;
   gpio_num_t busy;
   gpio_num_t clk;
   gpio_num_t mosi;
   int width;
   int height;
   bool initialized;
   spi_device_handle_t handle;
```

```
    int image_actualized_timestamp;
} Epd;
```

Výpis 11: Struktura pro E-Ink

Struktura Epd definuje:

- piny pro SPI
- frame_buffer - obsahující obrázek k vykreslení
- width - šířku E-Ink displeje
- height - výšku E-Ink displeje
- zda je E-Ink displej inicializován
- handle k SPI
- časové razítko (timestamp), které indikuje, poslední verzi obrázku

Každý pixel obrázku pro vykreslení, který se nachází v *frame_buffer*, je reprezentován jedním bitem, kde bit 1 je bílá barva a bit 0 černá. Pořadí pixelů je horizontální a v bytu je nejvíce významný bit první.

5 Implementace software pro základnu a čtečku

Pro vytvoření softwaru bylo využito vývojové prostředí Eclipse s podprojektem pro jazyk C/C++. Ke kompilaci a debugování byl použit dodaný program v Esp-Idf, který byl spouštěn přes aplikaci make. Při vývoji byl využit verzovací systém GIT s dvěma projekty (jeden pro modul základny a druhý pro modul čtečky).

Struktura zdrojových kódů a jejich použití:

- Společné zdrojové kódy
  - common.h
    - Obsahuje společné konstanty pro komunikaci a adresy funkcionalit
    - Obsahuje definici společných struktur a výčtů
    - Obsahuje hlavičky funkcí pro vytváření zpráv
  - processing_uart.h
    - Zajišťuje zpracování zprávy z UART
    - Zajišťuje zpracování chyb a potvrzování z UART
    - Obsahuje buffer pro příchozí data
  - sound.h
    - Zajišťuje funkce pro použití piezoměniče
    - Chráněno mutexem, aby nedocházelo k souběhu a hrála pouze jedna znělka v jednu chvíli.
  - uart.h
    - Zajišťuje posílání zpráv přes UART
    - Obsahuje funkci pro sestavení (parsování) zprávy z příchozích dat
  - utils.h
    - Obsahuje konfigurační funkce pro rozhraní

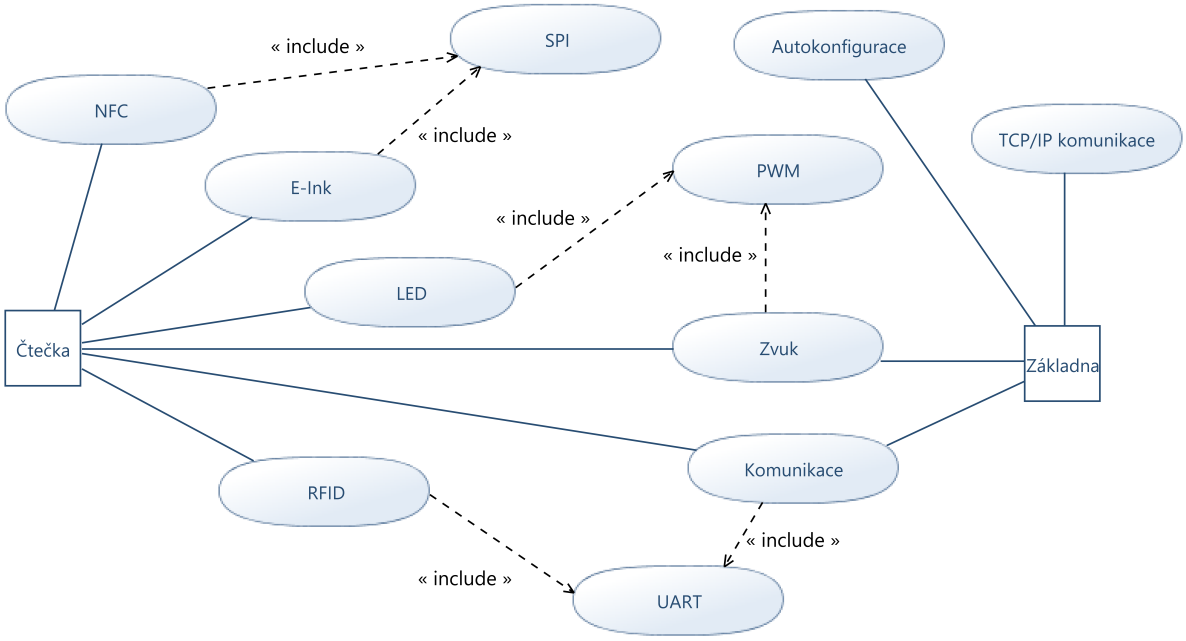
- Zdrojové kódy pro modul čtečky
  - default.h
    - Obsahuje výchozí obrázek pro E-Ink
  - eink_api.h
    - Obsahuje hlavičky funkcí pro práci s E-Ink displejem na vyšší úrovni
  - epd.h

    - Obsahuje ovládání E-Ink displeje
  - epdpaint.h
    - Zdrojový kód od firmy Waweshare dodávaný pro vykreslování textu a obrazců na E-Ink displej
  - fonts.h
    - Otevřený zdrojový kód od tvůrce MCD Application Team, který obsahuje definice fontů pro vykreslení na E-Ink displejích firmy Waweshare
  - lights.h
    - Obsahuje hlavičky funkcí pro ovládání LED diod
    - Zajišťuje indikaci chybového stavu
  - pn532EspIdf.h
    - Poskytuje rozhraní k ovládání PN532
  - reader_module_main.c
    - Vstupní bod programu
    - Obsahuje úlohy pro:
      - NFC
      - RFID
      - Úloha pro příjem zpráv a jejich zpracování

- Zdrojové kódy pro modul základny
  - dhcp.h
    - Obsahuje hlavičky funkcí pro zpracování JSON konfigurace a její interpretaci
    - Vytváří úlohy pro TCP/IP klienty
  - local_verification.h
    - Zajišťuje testovací funkcionalitu bez připojení k serverům, tedy karty se ověřují lokálně
  - tcp.h
    - Obsahuje hlavičky funkcí pro vytváření TCP/IP klientů
    - TCP/IP úlohy
    - Zpracování příchozích dat
      - V novém režimu
      - V kompatibilním režimu

    - Příjem dat a jejich poslání do výstupní fronty a případnou fragmentaci
- Ethernet_module.c
    - Vstupní bod programu
    - Obsahuje úloha pro příjem zpráv a jejich zpracování
    - Obsahuje úlohu pro přenos zpráv přes UART z výstupní fronty

Diagram, ukazující jaké funkcionality moduly využívají a na čem závisejí, je vyobrazen na obrázku č. 9.



Obrázek 9: Funkcionality modulů a jejich závislosti

5.1 Komunikační protokol mezi základnou a čtečkou

Komunikační protokol mezi modulem čtečky a základny je navržen pro sériovou linku. Je inspirován protokolem PPP a HDLC. Navržený protokol je orientovaný po bytech. Pro potvrzování zpráv se používá jednotlivé potvrzování (stop and wait), které bylo zvoleno pro svou jednoduchost implementace a nízkou latenci na sběrnici RS422. [29] Tento protokol je navržen tak, aby přechod na sofistikovanější systém potvrzování zpráv nebyl v budoucnu problematický. V případě poškozeného paketu se používá negativní potvrzení pro urychlení znovu zaslání paketu.

Paket ohraničují křídlové značky. Značka označující začátek paketu je 0x55. Po křídlové značce následuje dalších šest bitů adresy, označující, které periferii zpráva patří, případně také zda se jedná o potvrzovací paket. Dalších deset bitů je délka dat paketu. Poté následuje 1 byte označující číslo paketu, které je využíváno pro potřeby potvrzování. Dále následují byty s daty,

kterých je podle délky v hlavičce. Za daty následuje 16bitový kontrolní součet CRC16, který umožňuje zjistit, zda data jsou v pořádku. Paket ukončuje pak byte s hodnotou 0xAA.

Struktura paketu:

Tabulka 3: Struktura paketu

0x55	Adresa	Délka	Číslo paketu	Data	CRC16	0xAA
8 bitů	6 bitů	10 bitů	8 bitů	Délka × 8 bitů	16 bitů	8 bitů

Použité adresy a jejich význam:

Tabulka 4: Adresy a význam

Adresa	Význam
0b111111	Adresa označující potvrzení
0b000000	Adresa označující negativní potvrzení
0b000001	Adresa pro periferii RFID
0b000010	Adresa pro periferii NFC
0b000100	Adresa pro PWM ovládání LED diod
0b000110	Adresa pro ovládání piezoměniče
0b000111	Adresa pro obecné příkazy (restart, chybové stavy, ...)
0b000011	Sdružená adresa pro zvuk a LED diody
0b000101	Adresa pro E-Ink displej

Velikost objemu dat potřebných pro překreslení obrázku na E-Ink displeji je větší než velikost jednoho paketu. V takovém případě je první byte paketu využit jako číslo řádku v E-Ink displeji a řádek s hodnotou 255 je použit jako obecný příkaz pro E-Ink periferii.

Kontrolní součet CRC16 je cyklický redundantní součet o velikosti 16 bitů. Je to speciální hashovací funkce používaná k detekci chyb při přenosu nebo uložení dat. Kontrolní součet je odesílán spolu s paketem a poté nezávisle vypočítán z dat, která byla přijata. Pokud je stejný, tak je zaručena bezchybnost přenosu dat. Při výpočtu kontrolního součtu se využívá metoda dělení polynomů. [11] Byl využit typ výpočtu CRC-16-CCITT, který je například využit v technologii Bluetooth. [30]

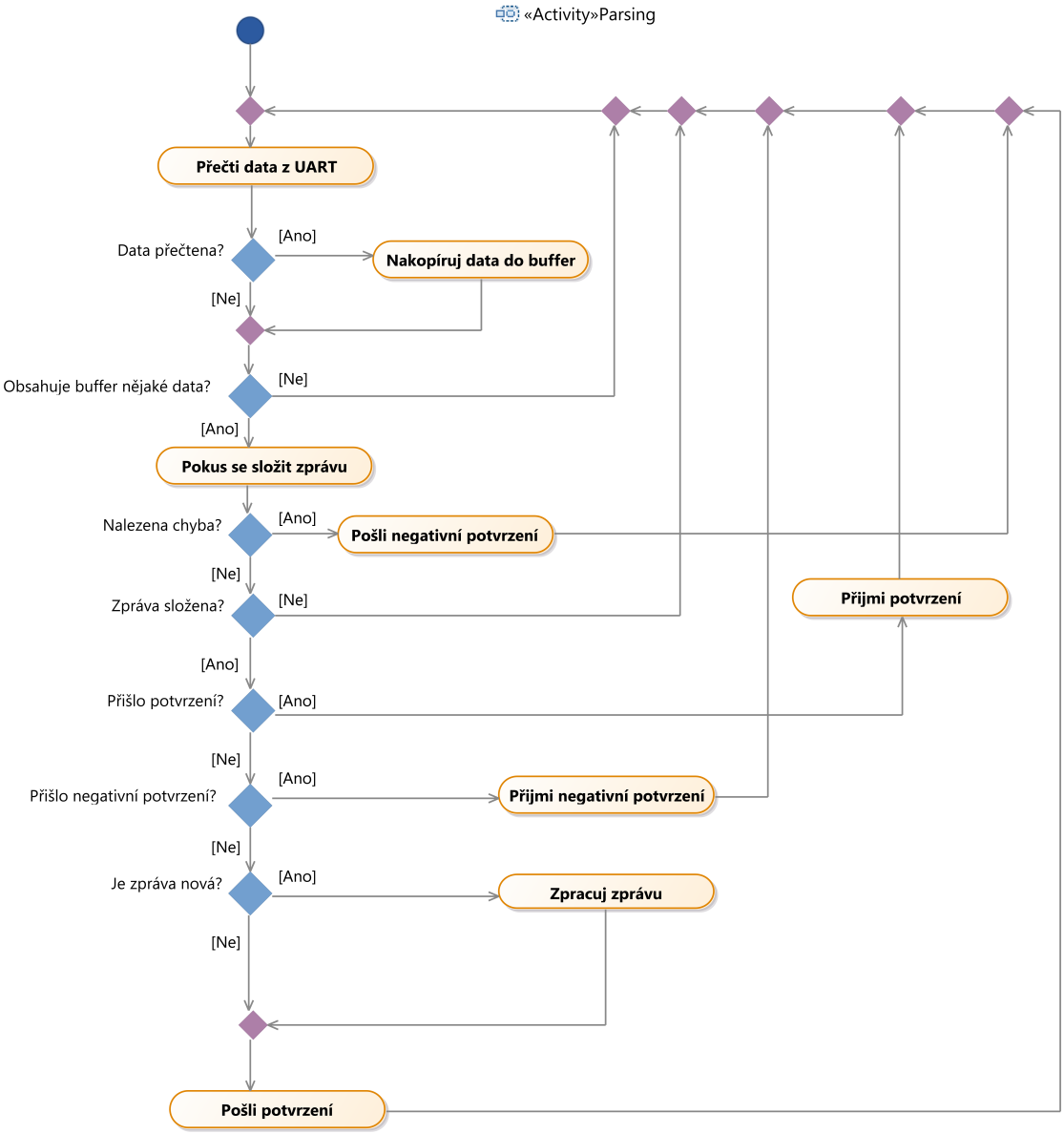
5.1.1 Zpracování zpráv

Zpracování zpráv zajišťuje funkce *process_uart_data*, která očekává nově přečtená data a jejich délku, a má dva výstupní parametry, a to pro zprávu a adresu. Návratová hodnota určuje, zda se zpráva má použít. Funkce obsahuje buffer pro data, z kterého další funkce *try_parse* se snaží sestavit jednotlivé zprávy. Funkce dále zajišťuje poslání negativního potvrzení v případě chyby, pozitivního v případě úspěšně došlé zprávy. Pokud zprávu zařízení nezná (určuje se podle čísla zprávy) a je bezchybná a celá, tak je předána dále.

Struktura zprávy, která je sestavená a předána dále k použití:

```
typedef struct {
   size_t len;
   uint8_t data[MAX_MSG_LEN];
   uint8_t packet_num;
} decoded_message_t;
```

Výpis 12: Struktura zprávy



Obrázek 10: Zpracování zprávy

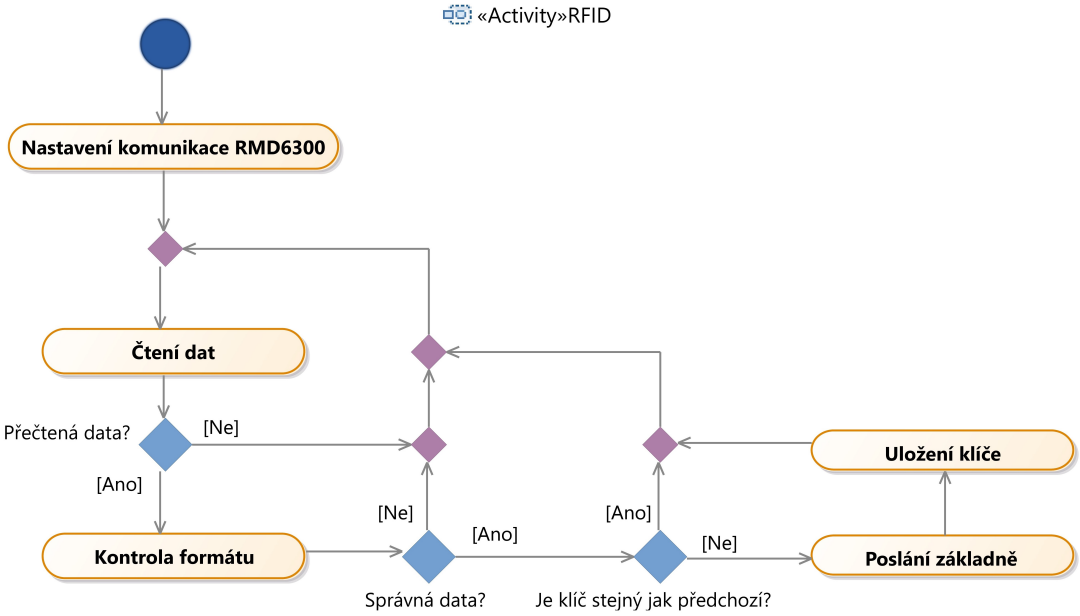
5.2 Popis software pro modul čtečky

Základní funkci čtečky obstarávají čtyři úlohy, které jsou spuštěny po startu a zůstávají spuštěny po celou dobu chodu čtečky:

- RFID úloha
- NFC úloha
- Úloha pro E-Ink displej
- UART úloha zajišťující příjem zpráv

5.2.1 RFID úloha

RFID úloha nastaví komunikaci s RMD6300 pomocí UART a nastaví příslušný pin pro napájení. Poté ve smyčce čeká na data z RMD6300. Data jsou identifikační číslo karty ve formátu typu EM4100. Pokud přijdou data, provede se jejich kontrola, zda jsou ve správném formátu a odpovídá kontrolní součet. Následně se porovná načtený identifikační klíč s identifikačním klíčem načteným v předchozím kroku. Jsou-li klíče identické, nejedná se o přiložení nové karty, a tedy data nejsou dále zpracována. Mazání předchozího klíče zajišťuje časovač, který při vypršení intervalu smaže předchozí identifikační data. Tento časovač je spuštěn při úspěšném přečtení karty. Porovnávání s předchozím klíčem se provádí z důvodu, že RMD6300 čte kartu kontinuálně a došlo by k nadměrnému posílání identifikačního klíče, který by byl čten vícekrát v krátkém časovém úseku. Identifikační klíč se pošle základně pomocí rozhraní UART a uloží se do předchozího identifikační klíče, aby nebyl zaslán opakovaně.

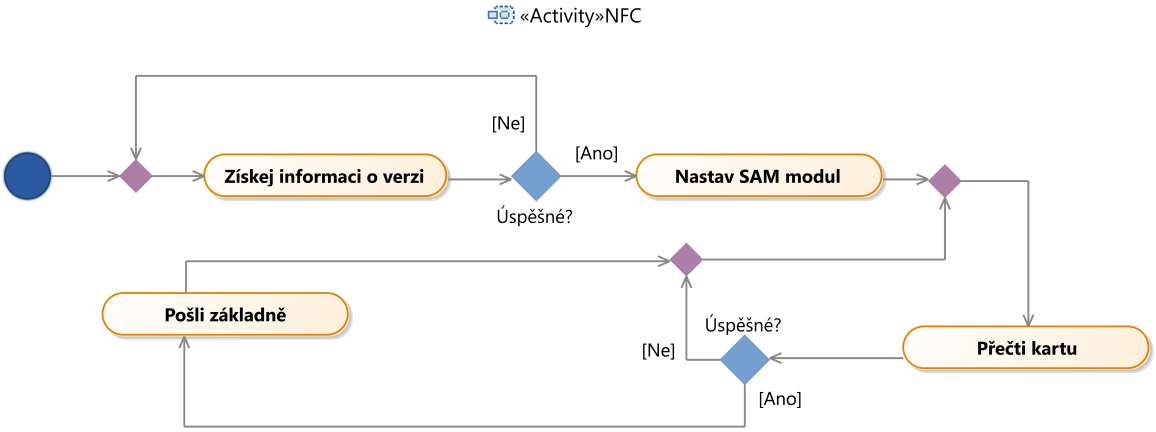


Obrázek 11: Zpracování dat z RMD6300

5.2.2 NFC úloha

NFC úloha pracuje podobně, jak RFID úloha s tím rozdílem, že se neporovnává předchozí karta s aktuální a to proto, že dané karty jsou při přiložení přečteny pomocí PN532 pouze jednou. Úloha inicializuje komunikaci s PN532 a poté periodicky ve smyčce čeká na čtení karty.

Pro použití PN532 se po nastavení SPI rozhraní pošle dotaz pro zjištění verze PN532. Ve chvíli, kdy pro tento dotaz přijde správná odpověď, tak se má PN532 za použitelné a přejde se ke konfiguraci SAM (secure access module) modulu. Pak je PN532 plně nastavené a lze použít.



Obrázek 12: Zpracování dat z PN532

5.2.3 E-Ink úloha

Úloha pro E-Ink displej inicializuje E-Ink displej, vykreslí výchozí obrázek a výchozí informace (verze, IP a MAC adresu) na displej a poté posílá v určitém časovém intervalu požadavek pro zaslání nového obrázku ze serveru. S požadavkem posílá také informaci o MAC adrese a čase, kdy nastala poslední známá aktualizace obrázku na serveru. Pokud server má novější verzi obrázku k vykreslení na E-Ink displej, tak ji pošle, jinak požadavek ignoruje. Při posílání nového obrázku jsou první čtyři byte časové razítko indikující, kdy nastala změna obrázku na serveru. Toto si čtečka uloží a posílá s každým novým požadavkem na obrázek. Takto je zamezeno, aby se obrázek k vykreslení posílal častěji, než je nutné.

Součástí softwarové obsluhy displeje je i úloha, která neběží po celou dobu běhu zařízení, ale je spouštěna v případě potřeby je úloha *refresh_eink_task*. Tato úloha vykreslí obsah bufferu pro E-Ink displej, který je v konfigurační struktuře E-Ink a poté přepne displej do sleep režimu. Tato úloha využívá mutex, aby nedošlo k souběhu, a to k takovému, že by byla tato úloha spuštěna vícekrát. To by mohlo nastat, například v případě, že přijde nový obrázek k vykreslení

a zároveň informace o chybě, která je potřeba vypsat. Tím by nastal souběh, který není v kódu ovládajícím E-Ink ošetřen.
Zdrojový kód pro překreslení displeje:

```
if (xSemaphoreTake( refresh_semaphore, ( TickType_t ) 1000 ) == pdTRUE) {
   Init(&epd);
   DisplayFrame(&epd, epd.frame_buffer);
   WaitUntilIdle(&epd);
   Sleep(&epd);
   xSemaphoreGive(refresh_semaphore);
}
```

Výpis 13: Překreslení displeje

5.2.4 UART úloha

Úloha pro UART komunikaci periodicky čte z UART portu data a snaží se je zpracovat na jednotlivé zprávy. Ke zpracování využívá funkci *process_uart_data*. Zpracovanou zprávu poté dle adresy předá dané funkcionalitě. Funkcionality, kterým jsou zprávy předávány, jsou tyto:

- E-Ink displej
- LED diody
- Zvuk – Piezoměnič
- Příkazy/Chybové stavy
- A jedna společná adresa pro LED diody a zvuk zároveň

```
while (1) {
   memset(data, 0, sizeof(RX_BUF_SIZE + 1));
   int rx_bytes = 0;
   if (buf_len < (sizeof(DATABUFFERLEN) - 200)) {
      rx_bytes = uart_read_bytes(UART_NUM_1, data, RX_BUF_SIZE,
            10 / portTICK_RATE_MS);
   }
   uint8_t address = 0;
   decoded_message_t parsed = { };
   if (process_uart_data(data, rx_bytes, &parsed, &address)) {
   ...
```

Výpis 14: Zpracování zprávy

5.2.5 Indikace stavu čtečky LED diodami

Pomocí LED diod na čtečce jsou indikovány chybové i informační stavy, ve kterých se čtečka může nacházet. V případě, že nastane potřeba indikace stavu nebo dojde zpráva do čtečky s informací o stavu základny, tak se spustí softwarový časovač, který zajišťuje střídání („blikání“) barev LED diod. Pokud potřeba indikace stavu nebo dojde zpráva, že stav je vyřešen, tak je tento časovač zastaven. Těmito stavy jsou:

- Chybové
  - Čtečka bez připojení k základně
  - Ztráta připojení k serveru
  - Chybný konfigurační JSON soubor
- Informační
  - Problém vyřešen

5.3 Přístupový server

Pro stávající řídící přístupový server je zachována kompatibilita pro nově navržený HW čtečky a základny včetně jejich SW. Současný přístupový sever na VŠB - Technické univerzitě Ostrava je napsán v jazyce C. Program využívá funkce *fork()* pro vytváření child procesů. Pro každou nově připojenou čtečku se vytvoří child proces, který se následně stará o komunikaci se čtečkou. Přístupový server očekává příjem dat s identifikačním číslem karty. Tato data jsou serverem zpracována a následně server odesílá čtečce informaci o povolení nebo zamítnutí vstupu.

Pro zachování kompatibility nového a starého systému je nutné data načtená ze čtečky RFID karet upravit tak, aby jejich formát odpovídal starému přístupovému systému.

Zdrojový kód pro úpravu dat pro zachování kompatibility se starým přístupovým systémem:

```
bool rfid_reader_out(decoded_message_t* msg_holder) {
   msg_holder->len += 2;
   msg_holder->data[15] = msg_holder->data[13];
   msg_holder->data[13] = 0x0d;
   msg_holder->data[14] = 0x0a;
   return true;
}
```

Výpis 15: Funkce upravující rfid data na starší formát

Ze serveru přicházejí data do základny ve formátu, kdy každá zpráva začíná bytem s hodnotou 0x27 a následuje byte specifikující typ zprávy. Po celou dobu komunikace serveru se základnou přichází data pro lcd displej, protože ten však nový přístupový systém nemá, tak jsou tato data

zahazována. Software pro nové čtečky také ignoruje příkazy pro ovládání podsvícení stávající čtečky. Nový přístupový systém zpracovává pouze příkaz typu „b“, který je interpretován jako odmítnutí přístupu a příkaz „o“, který je interpretován jako příkaz pro povolení přístupu. První příkaz typu „o“ otevře dveře, spustí oznamovací tóny a změní barvu podsvícení E-Ink displeje. Následující příkazy typu „o“ značí, jakou dobu má být spínací relé sepnuté. Tím se zachovává kompatibilita s fungováním původního systému, kde byla použita možnost ovlivnit, jak dlouho bude elektrický zámek dveří otevřen. Každý další příkaz typu „o“ prodlužuje dobu otevření elektrického zámku dveří o dalších 50 ms.

5.4 Popis software pro modul základny

Úlohy, které běží celou dobu programu, jsou dvě.

- příjem komunikace z čtečky
- pro odesílání zpráv čtečce.

5.4.1 Úloha pro příjem dat

Úloha pro příjem dat z čtečky pracuje podobně, jak obdobná úloha u čtečky s tím rozdílem, že adresy nejsou dány napevno pro dané funkcionality, ale každá adresa by měla odpovídat jedné frontě pro TCP/IP klienta. Toto mapování vzniká při konfiguraci sítě pomocí protokolu DHCP.

5.4.2 Úloha pro posílání zpráv

Úloha pro posílání zpráv čtečce periodicky získává zprávy ze společné fronty určené pro odesílání zpráv čtečce. Nastaví odpovídající číslo zprávy a zprávu odešle pomocí UART rozhraní. Použití fronty umožňuje určit prioritní data, která lze dávat na začátek fronty, a také je blokována pouze jedna úloha při posílání.

5.4.3 DHCP

Dynamic Host Configuration Protocol je síťový protokol z rodiny UDP/IP používaný pro automatickou konfiguraci zařízení při připojení do sítě. DHCP server dynamicky přiřazuje IP adresu a další síťové nastavení zařízením, tak aby zařízení mohla komunikovat v síti. DHCP server přiřazuje nastavení síťovým zařízením na určenou dobu. Po uplynutí této doby musí zařízení svůj dotaz na DHCP server opakovat. [31]
DHCP protokol má tyto operace:

Discovery – klient provede broadcast, že se připojil k dané síti

Offer – server nabídne IP adresu a nastavení

Request – klient požádá o nabízené nastavení

Acknowledge – server potvrdí nastavení

Protokol DHCP má také množství nastavení (options), která jsou specifikována v standardu RFC 2132. Každé nastavení zařízení má definovanou určitou číselnou hodnotu.

5.4.4 DHCP a konfigurace

Jakmile se na rozhraní Ethernetu objeví aktivní link, tak začne základna přístupového systému žádat o IP adresu DHCP server. V informacích o přiřazené adrese DHCP serverem zašle DHCP server také informace o tom, na kterém serveru se nachází další konfigurační informace a v dalším parametru je uvedena cesta ke konfiguračnímu souboru. Využívají se k tomu pole 66 - TFTP jméno serveru a pole 67 - název souboru, které jsou původně použité pro bootování ze sítě. [32] Po obdržení těchto informací základna zašle http požadavek na danou adresu. V požadavku je uvedena IP a MAC adrese základny pro účely identifikace základny. Tato identifikace umožní zasílání různých odlišných konfiguračních souborů pro různé základny.

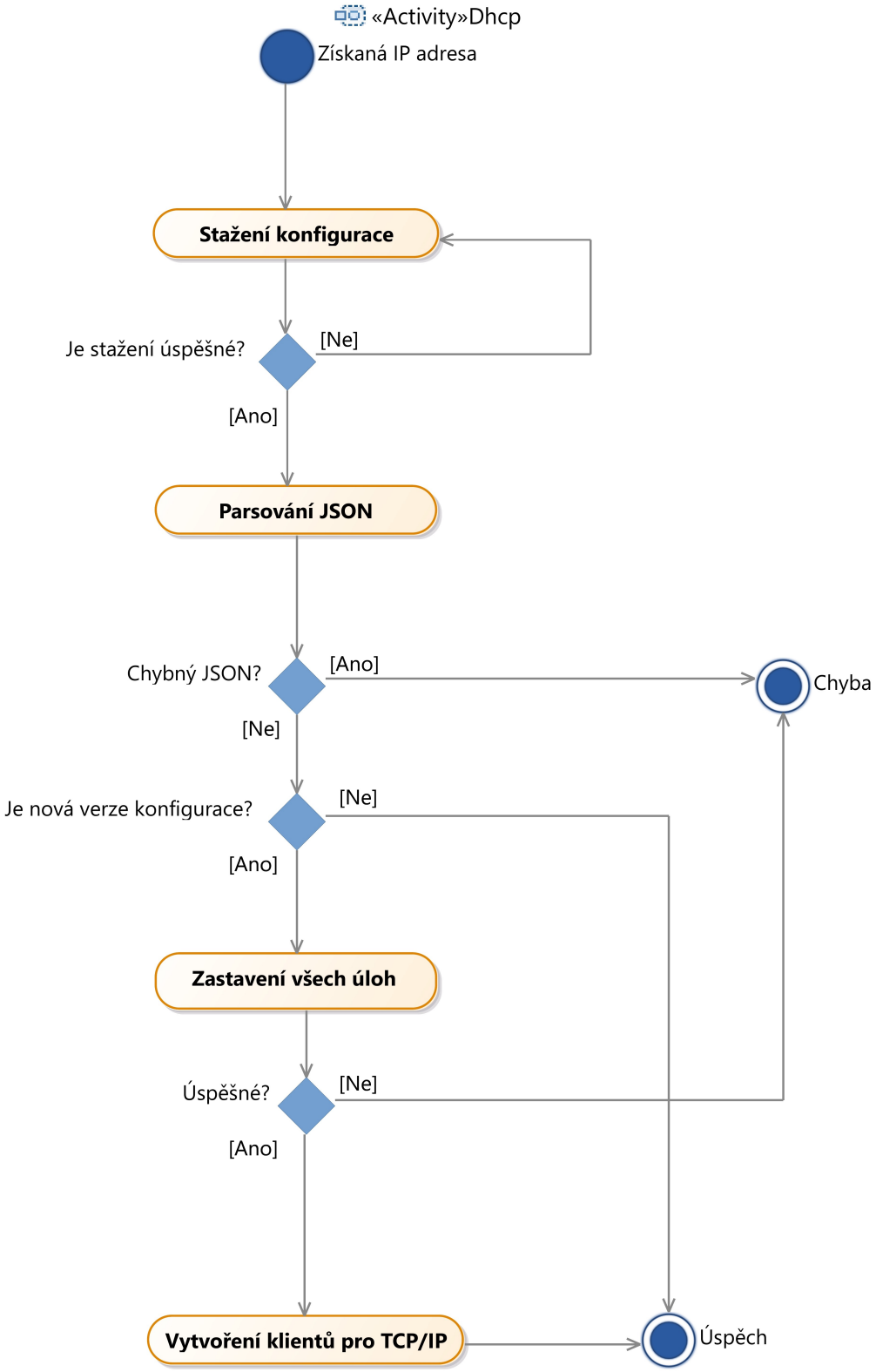
5.4.5 JSON

Konfigurační data jsou ve formátu JSON (JavaScript Object Notation), což je datový formát nezávislý na počítačové platformě, určený pro přenos dat a jeho výstupem je vždy řetězec. Formát JSON pochází z jazyka javascript. K jejich parsování se využívá otevřená knihovna cJson, která je dodávaná spolu s Esp-Idf.

Hodnota „version“ se porovná s verzí, kterou má základna aktuálně k dispozici, a podle této hodnoty se rozhodne, zda bude pokračovat se zpracováním zbytku načtené konfigurace. Pokud je na severu jiná verze, než má základna aktuálně k dispozici, jsou ukončeny všechny úlohy obsluhující sockety. Provede vytvoření nových úloh obsluhující sockety podle konfigurace, kde jsou informace o adrese periferie, adrese a portu obsluhujícího serveru a ostatních detailech.

```
{
   "Version":1, "SyslogEnabled":true,
   "SyslogServer":"ansfei.vsb.cz", "SyslogPort":63,
   "Settings":[
      {
         "QueueSize":4, "Legacy":true
         "Server":"ansfei.vsb.cz", "Port":10001,
         "Address":1, "Desc":"rfid",
         "In":"rfid", "Out":"rfid",
      }, ...
   ]
}
```

Výpis 16: Struktura konfiguračního JSON souboru



Obrázek 13: Aktivitní diagram konfigurace systému pomocí JSON

Tabulka 5: Základní prvky nastavení

Jméno nastavení	Popis	Datový typ
Version	Určuje verzi konfiguračního souboru. Základna podle této hodnoty pozná, zda je konfigurace novější, než má aktuálně	Číslo
SyslogEnabled	Určuje, zda se mají přesměrovat zprávy z UART na syslog server pomocí UDP.	Boolean
SyslogServer	Adresa serveru, na který se mají zasílat logovací zprávy pro syslog	Řetězec
SyslogPort	Port, na kterém přijímá syslog server zprávy	Číslo
Settings	Pole objektů, které obsahují jednotlivé konfigurace pro jednotlivé TCP/IP klienty.	Pole objektů

Tabulka 6: Vysvětlení objektu pro konfiguraci TCP/IP klientů

Jméno nastavení	Popis	Datový typ
QueueSize	Velikost fronty pro výstupní data, které odcházejí ze základny na server	Číslo
Desc	Popis nastavení. Pouze pro uživatele jako popisek. Program toto pole ignoruje	Řetězec
Server	Adresa serveru, na který má TCP/IP klient navázat spojení	Řetězec
Port	Číslo portu, na který se má připojit	Číslo
In	Jméno funkce, která má upravovat data přicházející ze serveru	Řetězec
Out	Jméno funkce, která má upravovat data přicházející ze čtečky	Řetězec
Legacy	Určuje, zda vstupní data se mají očekávat ve formátu kompatibilním se starým přístupovým systémem.	Boolean
Address	Adresa určující, jaká funkcionalita čtečky má být daným TCP/IP klientem obsluhována.	Číslo

5.4.6 Konfigurace pro zachování kompatibility se starým systémem

Konfigurace také obsahuje informaci, zda daná periferie funguje v *„legacy“* (kompatibilním se starým přístupovým systémem) režimu. To je použito pro komunikaci s přístupovým serverem obsluhujícím stávající RFID a tímto způsobem je zachována kompatibilita s tímto serverem. Tento server má jiný způsob zasílání zpráv a je zde také nutnost ignorovat některé zprávy (data pro LCD displej).

5.4.7 Úlohy pro komunikaci s TCP/IP servery

Další úlohy jsou vytvářeny dynamicky na základě přijaté konfigurace. Tyto úlohy mají na starost komunikaci se servery. Každá úloha naváže jedno spojení s určeným serverem a toto spojení udržuje.

Úloha čte ze své fronty zpráv zprávy přicházející ze čtečky a posílá je na daný server. Pokud je určen konfigurací delegát (ukazatel na funkci), tak ten je nad zprávou použit. Delegát danou zprávu upraví a jako návratovou hodnotu dá vědět, zda se zpráva má poslat na server. Toho se využívá například u klienta pro RFID, který upravuje data z RFID čtečky tak, aby byla kompatibilní s původním formátem dat.

Úloha také čte ze socketu přijímaná data a přeposílá je do společné fronty pro výstup na čtečku. V kompatibilním režimu, který je určen parametrem *„legacy“* v konfiguraci, očekává data ve formátu první byte je 0x27, druhý byte je typ zprávy a zajišťuje otevírání dveří a indikace informací na čtečce. V normálním režimu přichází první 4 byte informace o délce zprávy a poté následují data. Pokud SW obsluhující danou periferii očekává, že má příchozí data fragmentovat základna a ne server, tak základna data, která jsou větší než maximální délka, rozdělí na více zpráv a první byte dat je číslo fragmentu. Tudíž existuje omezení na maximální délku zprávy ze serveru. Toto omezení je 25 245 B (maximální délka dat jedné zprávy krát maximální počet fragmentů a tedy 99 krát 255), což je v současné době více než dostačující.

5.5 Logování v ESP32

V Esp-Idf je připraveno několik funkcí pro logování zpráv. Zprávy jdou filtrovat jak po čas běhu, tak v době kompilace. Každý modul má svůj tag (štítek), který ho identifikuje. Ve výchozím stavu se logy v Esp-Idf zasílají na sériový port. Úrovně logování jsou (od nejvyšší závažnosti po nejnižší):

- Error – Chyba
- Warning – Varování
- Info – Informace
- Debug – Ladění
- Verbose – Podrobné

5.6 Syslog

Syslog, který vznik v osmdesátých letech dvacátého století, je protokol pro záznam programových zpráv typu klient/server. Postupem času se stal standardním loggovacím protokolem pro Unix-like systémy. Využívá port 514 a zprávy mohou být zasílány pomocí UDP nebo TCP. Zprávy jsou v textovém formátu. [33]

Existují dvě specifikace:

- IETF – RFC5424 [34]
  - Novější, definovaný v roce 2009
  - Zprávy jsou kódovány v UTF-8
  - Formát: <PRI>VER TIMESTAMP HOSTNAME APP-NAME PROCID MSGID [SOURCETYPE@NM_IANA key1="val1"key2="val2"etc.]
- BSD – RFC3164 [35]
  - Nespecifikuje kódování zpráv ani další parametry
  - Starší, specifikoval status quo v dané době
  - Formát: <PRI>TIMESTAMP HOSTNAME APP-NAME[PROCID]: sourcetype="SOURCETYPE "key1="val1"key2="val2"etc.

5.6.1 Logování do syslogu

Pro modul základny jsou připraveny funkce, které umožňují přesměrovat logování z UART na daný syslog server. Toto nastavení se aktivuje pomocí konfigurace v JSON, kde jsou také určeny na který server a port se mají zprávy zasílat. Posílají se UDP pakety ve formát IETF, který je definovaný v RFC5424.

V Esp-Idf je možné nahradit standardní logovací funkcí, vlastní funkcí. Při získání IP adresy se zavolá funkce *udp_logging_init*, která vytvoří socket na syslog server, nahradí funkce pro výstup logu svou funkcí *udp_logging_vprintf*. Tato funkce připraví logovací zprávu do formátu, který očekává syslog server.

Při ztrátě konektivy je zavolána funkce *udp_logging_free*, která uzavře a uvolní socket a přesměruje logování zpět na UART.

```
int facility = 1 * 8;
int additional_len = sprintf((char *) buf, "<%d>1 - %s ESP32AccessSystem - %
    d - - ",
      severity + facility, hostname, proc_id);
char *cur_task = pcTaskGetTaskName(xTaskGetCurrentTaskHandle());
if (strncmp(cur_task, "tiT", 16) != 0) { //not LWIP TCP/IP task
   int len = vsprintf((char*) buf + additional_len, str, list) +
       additional_len;
   if ((err = sendto(udp_log_fd, buf, len, 0, (struct sockaddr *) &
       serveraddr,
         sizeof(serveraddr))) < 0) {
```

Výpis 17: Vytvoření a poslání syslog zprávy

5.7 NFC server

Server pro NFC je napsaný v jazyce C++. Uživatel si může zvolit, zda využije vláken nebo procesů pomocí *fork()*. Ve smyčce se čeká na příchozí připojení. Při novém připojení se vytvoří nové vlákno/proces, které bude obsluhovat dané připojení. Vlákno/proces čte data ze socketu a vypisuje na standardní výstup identifikační číslo karty NFC zaslané ze čtečky, ke které byla daná karta přiložena. Skládá se ze tří souborů:

exceptions.cpp – definice výjimek a loggování

Main.cpp – vstupní bod a hlavní část programu, obsahuje spouštění klientů, nápovědu a parsování spouštěcích parametrů

TcpSocket.cpp – obaluje socket do C++ třídy, která podporuje RAII a ošetřuje chybové stavy.

5.8 E-Ink server

Server pro získávání obrázků pro E-Ink displej je napsaný v jazyce Python. Na serveru se musí řešit rozdílné pořadí byte a práci s jednotlivými byte. K tomu se využívá standardního modulu struct. Jedno vlákno má na starost jednoho klienta a tvoří se při novém připojení klienta na server. Klient periodicky posílá požadavek pro nový obrázek. V tomto požadavku je také MAC adresa a čas poslední změny obrázku známého (vykresleného) pro čtečku. Podle MAC adresy server vybere správný obrázek, případně pokud takový neexistuje, tak se zvolí výchozí (default.bin). U vybraného obrázku se zjistí čas poslední změny a porovná se s přijatým. Pokud se liší, tak server pošle nový čas změny a poté obrázek. Celá komunikace se základnou funguje na principu, že se první pošle délka dat k odeslání a poté data.

5.9 Výzvy do budoucna a možné další rozšíření programu

Jednou z hlavních výzev do budoucna je připravit přístupový systém na čtení nových karet NFC, které se budou používat v nadcházejících letech na VŠB - Technické univerzitě Ostrava. Nyní není známo, jakým způsobem a jaký typ karet s podporou NFC bude zvolen a použit. Základní součásti použití NFC karet jsou však pokryty a implementace je připravena na rozšíření, takže by to neměl být zásadní problém.

Zajímavou možností je využít přístupové terminály pro lokalizaci osob, neboť ESP32 obsahuje rovněž Bluetooth vysílač. Ten by se dal využít jako Bluetooth majáček (beacon) a klientské zařízení by pomocí přibližného určení vzdálenosti od jednotlivých majáčků mohlo určit svou přibližnou polohu.

6 Testování

Pro testování a vývoj SW byly použity dvě vývojové sady. První sada byla sestavena z volně prodejných součástek. Další vývoj pak byl prováděn už na testovacím HW přímo navrženém pro nový přístupový systém. První testování probíhalo pouze s lokálním ověřováním karet. Tento styl ověřování lze stále ve zdrojovém kódu nastavit a použít.

Další testování SW probíhalo již s funkčním rozhraním pro Ethernet a virtuálním počítačem, ve kterém byly dostupné servery potřebné pro provoz systému. Hlavní testování pro ověření základní funkčnosti bylo prováděno na budově Nová FEI, VŠB - Technická univerzita Ostrava v prostorách katedry elektroenergetiky. Zde bylo postupně instalováno 14 sad nového přístupového systému. Toto testování probíhalo po dobu sedmi týdnů a pak byl systém uveden do reálného provozu. Instalované sady měly upraveny HW. Úpravy se týkaly ovládání LED diod a ovládání napájení modulu RFID. Tyto změny si vyžádaly i změny v programu čtečky.

Z výsledků tohoto hlavního testování byly vyvozeny určité potřebné změny ve fungování SW. Jednalo se o akustickou signalizaci při zapnutí a také při získání IP adresy. Tato funkcionalita byla upravena tak, že akustická signalizace proběhne pouze při zapnutí po výpadku napájení a pouze při prvním získání IP adresy. Další změnou, která vyplynula z testování, byla potřeba mít možnost nastavit délku sepnutí relé. Tento požadavek byl zpracován tak, aby vše bylo kompatibilní s předchozím systémem.

Vylepšila se také indikace chybových stavů a informace předávané uživateli po startu systému a získání IP adresy. A to tak, že nyní se informace vykreslují také na E-Ink displej, což je uživatelsky značně pohodlnější a velmi to urychluje identifikaci čteček při montáži. Probíhaly také určité úpravy a opravy v kódu pro zpracování zpráv z UART a také změny přenosové rychlosti na RS422. Použitá sériová komunikace RS422 se ukázala na dané vzdálenosti být bezchybná, takže se mohla použít vyšší rychlost přenosu, a to 115 200 Bd. Úpravám se nevyhnul ani SW pro E-Ink server, kde bylo přidáno ošetření výskytu chybných příchozích dat a ukončení spojení s klienty.

V dubnu 2019 bylo nainstalováno na budově Nová FEI, VŠB - Technická univerzita Ostrava 15 sad tohoto nového přístupového systému. Na katedře elektroenergetiky je instalováno 14 sad a jedna sada na katedře aplikované matematiky. Ke vstupu do učeben se prozatím používají stávající karty založené na technologii RFID. Funkcionalita karet s technologii NFC byla otestována, ale v současné době nejsou tyto karty na VŠB - Technická univerzita Ostrava používány. Celý systém je stabilní a funkční.

7 Závěr

V rámci této bakalářské práce je vyvinut software pro modul čtečky a základny na platformě poskytnutého HW nového přístupového systému a řešena jeho problematika. V teoretické části jsou vysvětleny použité technologie a rozhraní. Práce se zabývala také vývojem software na platformě Esp-Idf pro mikrokontroler ESP32. Dále byl vyvinut SW pro server obsluhující E-Ink displeje a server pro výpis přiložených karet s technologií NFC.

Je zachována kompatibilita s původním přístupovým systémem. Modul čtečky umožňuje jak použití RFID karet, tak i použití karet podporujících NFC. Při přechodu na použití přístupových karet s technologií NFC bude přístupový systém výrazně bezpečnější. Zdrojový kód je napsán tak, aby nebylo obtížné přidat novou funkcionalitu nebo pozměnit stávající. V modulu nové čtečky je nově využit E-Ink displej. Systém umožňuje vzdálenou konfiguraci prostřednictvím DHCP protokolu. Nově vyvinutý komunikační protokol je plně funkční a ověřen v praxi.

Tato práce byla pro mě obohacením, jak pro vývoj aplikací pro vestavěné systémy, tak v oblasti hardwaru a jeho použití. Získal jsem zkušenosti s prací s moderními mikroprocesory a vylepšil si znalost programovacího jazyka C. Pochopil jsem, jak fungují různá rozhraní a jak je využít. Seznámil jsem se s technologiemi NFC a RFID, s jejich vlastnostmi a historickým vývojem. Musel jsem si detailně nastudovat příslušné standardy. Využil jsem jak příslušných dokumentací, tak jsem ale i musel nahlédnout do vnitřních kódů knihoven, abych lépe pochopil jejich fungování.

Literatura

[1] *Esp32* [online]. [cit. 2019-04-07]. Dostupné z: http://esp32.net/

[2] ESPRESSIF SYSTEMS. *ESP32 Technical Reference Manual*. 4. 2018. Dostupné také z: https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf

[3] *Kolban's book on ESP32: The definitive guide to programming...* [online]. [cit. 2019-04-07]. ISBN ESP. Dostupné z: https://leanpub.com/kolban-ESP32

[4] *Arduino* [online]. [cit. 2019-04-07]. Dostupné z: https://www.arduino.cc/

[5] *Esp-Idf dokumentace* [online]. [cit. 2019-04-07]. Dostupné z: https://docs.espressif.com/projects/esp-idf/en/latest/

[6] *Eclipse IDE* [online]. [cit. 2019-04-08]. Dostupné z: https://docs.espressif.com/projects/esp-idf/en/stable/get-started/eclipse-setup.html

[7] *FreeRTOS* [online]. [cit. 2019-04-07]. Dostupné z: https://www.freertos.org

[8] *ESP-IDF FreeRTOS SMP Changes* [online]. [cit. 2019-04-08]. Dostupné z: https://docs.espressif.com/projects/esp-idf/en/stable/api-guides/freertos-smp.html

[9] TIŠNOVSKÝ, Pavel. Sběrnice RS-422, RS-423 a RS-485. *Root* [online]. [cit. 2019-04-07]. Dostupné z: https://www.root.cz/clanky/sbernice-rs-422-rs-423-a-rs-485/

[10] BIDGOLI, Hossein. *Handbook of computer networks: key concepts, data transmission, and digital and optical networks* [online]. Hoboken, N.J., c2008 [cit. 2019-04-07]. ISBN 978-047-1784-609.

[11] TANENBAUM, Andrew S. *Computer networks*. 4th ed. Upper Saddle River, NJ: Prentice Hall PTR, c2003. ISBN 01-306-6102-3.

[12] RS-232. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2019-04-07]. Dostupné z: https://cs.wikipedia.org/wiki/RS-232

[13] RS-422. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2019-04-07]. Dostupné z: https://cs.wikipedia.org/wiki/RS-422

[14] SPI. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2019-04-08]. Dostupné z: https://en.wikipedia.org/wiki/Serial_Peripheral_Interface

[15] SEN, Dipankar, Prosenjit SEN a Anand M. DAS. *RFID for Energy & Utility Industries*. 1. Tulsa, Okla.: PennWell, c2009. ISBN 978-1593701055.

[16] A. WEIS, Stephen. *RFID (Radio Frequency Identification) : Principles and Applications* [online]. 2007 [cit. 2019-04-07]. Dostupné z: https://www.semanticscholar.org/paper/RFID-(-Radio-Frequency-Identification-)-%3A-and-Weis/aaa32dc1aa5539198693c43c5e99ab6c61b9e356. MIT CSAIL.

[17] *RMD6300 specifikace.* Dostupné také z:https://github.com/SeeedDocument/125Khz_RFID_module-UART/raw/master/res/RDM630-Spec.pdf

[18] HUBÁČEK, Bc. Stanislav. *Bezpečnostní audit RFID systémů.* Zlín, 2011. Univerzita Tomáše Bati ve Zlíně.

[19] VANĚK, T., HOLENDA a ROHLÍK. *Klonování RFID čipů na přístupových kartách* [online]. České vysoké učení technické v Praze, FEL, 30. 07. 2012 [cit. 2019-04-07]. Dostupné z: http://access.fel.cvut.cz/view.php?nazevclanku=klonovani-rfid-cipu-na-pristupovych-kartach&cisloclanku=2012070003

[20] FINKENZELLER, Klaus. *Fundamentals and applications in contactless smart cards, radio frequency identification and near-field communication.* 3rd ed. Hoboken, NJ: Wiley, c2010. ISBN 978-0-470-69506-7.

[21] ISO/IEC 7816-4:2005. *Identification cards – Integrated circuit cards – Part 4: Organization, security and commands for interchange.* 2. ISO copyright office, 2005-01.

[22] *PN532 User Manual.* 2. NXP. UM0701-02. Dostupné také z: https://www.nxp.com/docs/en/user-guide/141520.pdf

[23] Pulzně šířková modulace. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2019-04-07]. Dostupné z: https://cs.wikipedia.org/wiki/Pulzn%C4%9B_%C5%A1%C3%AD%C5%99kov%C3%A1_modulace

[24] *Https://electronics.stackexchange.com/questions/360390/what-is-the-difference-between-mcpwm-and-ledc-pwm* [online]. In: . [cit. 2019-04-07].

[25] Electronic paper. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2019-04-07]. Dostupné z: https://en.wikipedia.org/wiki/Electronic_paper#Technologies

[26] *Waveshare* [online]. [cit. 2019-04-07]. Dostupné z: https://www.waveshare.com/product/modules/oleds-lcds/e-paper.htm

[27] OLIVKA, Ing. Petr. *Zobrazovací jednotky počítačů - monitory: Studijní materiál pro předmět Architektury počítačů* [online]. 2011 [cit. 2019-04-07]. Dostupné z: http://poli.cs.vsb.cz/edu/apps/down/monitory.pdf

[28] *2.7inch e-Paper HA* [online]. 2. 2018.12.05 [cit. 2019-04-07]. Dostupné z: https://www.waveshare.com/w/upload/3/31/2.7inch_e-paper_hat_user_manual_en.pdf

[29] Cyclic redundancy check. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2019-04-07]. Dostupné z: https://en.wikipedia.org/wiki/Cyclic_redundancy_check

[30] , Mats, NASLUND, Goran SELANDER a Fredrik LINDQVIST. *Cryptographically Secure CRC for Lightweight Message Authentication* [online]. [cit. 2019-04-07]. Dostupné z: https://eprint.iacr.org/2015/035.pdf. Royal Institute of Technology, 164 40 Stockholm, Sweden.

[31] Dynamic Host Configuration Protocol. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2019-04-10]. Dostupné z: https://cs.wikipedia.org/wiki/Dynamic_Host_Configuration_Protocol

[32] *DHCP Options and BOOTP Vendor Extensions.* rfc2132. Dostupné také z: https://tools.ietf.org/html/rfc2132

[33] Syslog. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2019-04-08]. Dostupné z: https://cs.wikipedia.org/wiki/Syslog

[34] *The Syslog Protocol.* Network Working Group, 2009. rfc5424. Dostupné také z: https://tools.ietf.org/html/rfc5424

[35] RFC3164. *The BSD syslog Protocol.* Network Working Group, 2001.

[36] MSYS2, *Toolchain 20181001* [online]. [cit. 2019-04-08]. Dostupné z: https://dl.espressif.com/dl/esp32_win32_msys2_environment_and_toolchain-20181001.zip

A Struktura přiložených souborů

Přiložené soubory obsahují servery pro E-Ink a NFC a dva projekty pro ESP32. Jeden je pro modul čtečky a druhý pro modul základny. Dále také obsahuje celé Esp-Idf a to z důvodu, že novější verze nemusí být kompatibilní s napsaným zdrojovým kódem. Popis složek a souborů:

esp-idf – složka s Esp-Idf potřebná ke kompilaci. Jsou z ní odstraněny příklady (Examples)

pristupovy-system – Obsahuje data pro modul základny

- **main** – Obsahuje zdrojové kódy pro modul základny
- **server** – Obsahuje server pro E-Ink
- **server_certs** – Certifikáty pro SSL
- **build** – Do této složky se kompilují zdrojové kódy

reader_module – Obsahuje data pro modul čtečky

- **main** – Obsahuje zdrojové kódy pro modul čtečky
- **build** – Do této složky se kompilují zdrojové kódy

nfc_server – Obsahuje zdrojové kódy pro NFC server

B Kompilace pro Esp-Idf ve Windows

Ke kompilaci projektů v prostředí Windows potřebujeme GNU-kompatibilní prostředí. Dokumentace Esp-Idf doporučuje MSYS2.

B.1 Stažení nástrojů

K instalaci MSYS2 a ostatních potřebných nástrojů nabízí Esp-Idf předpřipravený zip archív.[36] Ten stačí pouze rozbalit a je připraven k použití. Po zbytek textu se předpokládá, že archív byl rozbalen do cesty `C:\` .

B.2 Získání Esp-Idf

Dalším krokem je instalace Esp-Idf. Pro zprovoznění této bakalářské práce stačí zkopírovat přílohu do cesty `C:\msys32\home\user\esp` . Pro stažení nové verze lze využít terminál z MSYS2 a zadat tyto příkazy:

```
cd ~/esp
git clone --recursive "https://github.com/espressif/esp-idf.git"
```

Výpis 18: Stažení Esp-Idf

B.3 Přidání proměnné IDF_PATH do systému

Pro přidání proměnné do systému MSYS2 musíme přidat soubor `export_idf_path.sh` do cesty `C:/msys32/etc/profile.d/` s příkazem `export IDF_PATH="C:/msys32/home/user-name/esp/esp-idf"`, kde cesta je cesta ke složce `esp_idf`.

B.4 Instalace Python balíčků

Esp-Idf vyžaduje Python balíčky. Potřebné balíčky jsou definovány v `IDF_PATH/requirements.txt`. Nainstalovat je jde pomocí příkazu `python -m pip install --user -r $IDF_PATH/requirements.txt`.

B.5 Použití Esp-Idf

Po připojení zařízení a zjištění portu, na kterém je zařízení připojeno, se musí spustit konfigurace pomocí `make menuconfig`. Po nastavení seriového portu v této konfiguraci lze již přiložené projekty zkompilovat a nahrát do zařízení pomocí příkazu `make flash`.

B.6 Monitor

Pomocí příkazu `make monitor` můžeme sledovat log ze zařízení a případně zařízení restartovat a další možnosti.